7

Generalized Association Rule Mining Using Genetic Algorithms

Peter P. Wakabi-Waiswa, Venansius Baryamureeba and K. Sarukesi

We formulate a general Association rule mining model for extracting useful information from very large databases. An interactive Association rule mining system is designed using a combination of genetic algorithms and a modified a-priori based algorithm. The association rule mining problem is modeled as a multi-objective combinatorial problem which is solved using genetic algorithms. The combination of genetic algorithms with a-priori query optimization make association rule mining yield fast results. In this paper we use the same combination to extend it to a much more general context allowing efficient mining of very large databases for many different kinds of patterns. Given a large database of transactions, where each transaction consists of a set of items, and a taxonomy (is-a hierarchy) on the items, we find associations between items at any level of the taxonomy. We show how the idea can be used either in a general purpose mining system or in a next generation of conventional query optimizers.

1. Introduction

Association rule mining (ARM) is one of the core data mining techniques. The major aim of ARM is to find the set of all subsets of items or attributes that frequently occur in many database records or transactions, and additionally, to extract rules on how a subset of items influences the presence of another subset. The ARM problem was introduced in 1993 by Agrawal et al. [1993] who developed the Apriori algorithm for solving the ARM problem. Most of the existing algorithms are improvements to Apriori algorithm [Ghosh and Nath, 2004; Zhao and Bhowmick, 2003].

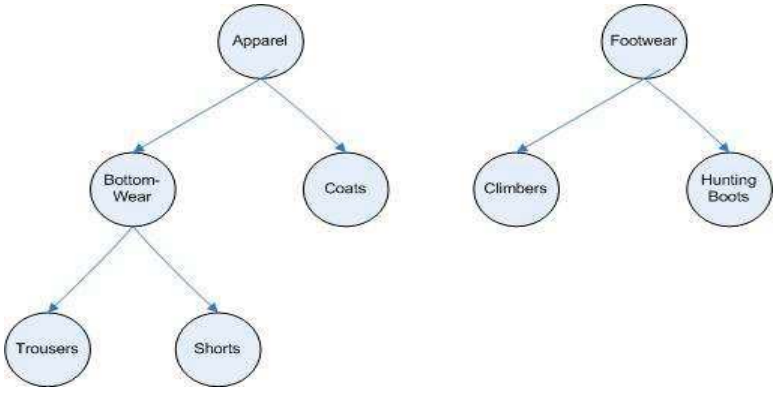
Given a set of transactions, where each transaction is a set of items, an association rule is an expression of the form $X \Rightarrow Y$ where X and Y are sets of items. This rule translates to stating that transactions in the database which contain the items in X tend to contain the items in Y. An example of such a rule might be that 20% of transactions that contain beer also contain diapers; 3% of all transactions contain both these items". Here 20% is called the confidence of the rule, and 3% the support of the rule. The support of the rule $X \Rightarrow Y$ is the percentage of transactions that contain both X and Y. The problem of mining association rules is to find all rules that satisfy a user-specified minimum support and minimum confidence.

In ARM, frequent pattern mining (FPM) is the most time-consuming part. The traditional association rule mining algorithms that extract all frequent generalized patterns are not very efficient due to the fact that a large portion of the frequent patterns are redundant. This subjects users to a drawback in that when faced with real problems they tend to get a lot of uninteresting or redundant rules along with the interesting rules. During the mining task, in most cases, taxonomies (is-a hierarchies) do exist over the items being mined.

Therefore association rules can be extracted in a generalized form conveying knowledge in a compact manner with the use of taxonomies. According to Srikant and Agrawal (1995) [10], given a large database of transactions, where each transaction consists of a set of items, and a taxonomy (is-a hierarchy) on the items, generalized association mining involves finding associations between items at any level of the taxonomy. Figure 1 shows an example of a taxonomy. The taxonomy says that trousers is-a bottom wear, shorts is-a bottom wear, bottom wear is-a Apparel. The motivation for generalized association mining is that users are interested in generating rules that span different levels of the taxonomy. In generalised association rules, application specific knowledge in the form of taxonomies over items are used to discover more interesting rules.

In this paper, we formulate a mechanism for finding for generalized association rules. This is achieved as a combination of an a – priori based algorithm for finding frequent itemsets with multi-objective evolutionary algorithms (MOEA) with emphasis on genetic algorithms (GA). The main motivation for using GAs is that they perform a global search and cope better with attribute interaction than the greedy rule induction algorithms often used in data mining tasks. We emphasize generalized association rule mining because users may be interested in generating rules that span different levels of the taxonomy. Multi-objective optimisation with evolutionary algorithms is well discussed by (Fonseca and Fleming, 1998) [4], (Khabzaoui et al. 2005) [8] and (Freitas, 2003 [5]).

Fig.1: Example of is a Taxonomy



$L = \{i_1, \ldots, i_m\}$ denotes a set of literals, also referred to as itemsets. T denotes

a directed acyclic graph (DAG) on L and it is a set of taxonomies. An edge in T represents an "is-a" relationship, and T represents a set of taxonomies. The taxonomy is modelled as a DAG rather than a forest to allow for multiple taxonomies. Lower case letters denote items and upper case letters denote itemsets. $x^\wedge$ denotes an ancestor of x (and x a descendant of $x^\wedge$) if there is an edge from $x^\wedge$ to x in the transitive-closure of T. Note that a node is not an ancestor of itself, since the graph is acyclic. D denotes a set of transactions, where each transaction T is a set of items such that $\mathsf{T} \subseteq \mathsf{L}$. A transaction T is said to support an item $x \in \mathsf{L}$ if x is in T or x is an ancestor of some item in T. We say that a transaction T supports $x \in \mathsf{L}$ if T supports every item in X.

A generalized association rule is an implication of the form $X \Rightarrow Y$, where $X \subset \mathsf{L}$, $Y \subset \mathsf{L}$, $X \bigcap Y = \varnothing$, and no item in Y is an ancestor of any item in X. The rule $X \Rightarrow Y$, holds in the transaction set D with confidence c if c% of transactions in D that support X also support Y. The rule $X \Rightarrow Y$, has support s in the transaction set D if s% of transactions in D support $X \bigcup Y$. The reason for the condition that no item in Y should be an ancestor of any item in X is that a rule of the form $x \Rightarrow ancestor(x)$ is trivially true with 100% confidence, and hence redundant Agrawal (1995) [10]. We call these rules generalized association rules because both X and Y can contain items from any level of taxonomy T

The rest of this paper is organised as follows. In Section 2 we provide an overview of work related to mining generalised association rules. In Section 3 we discuss the proposed algorithm. In Section 4, we present an overview of multi-objective optimisation and rule mining problems . In Section 5 the analysis of the results are presented. Section 6 is the conclusion.

2. Related Literature

According to Srikant and Agrawal (1995) [10], the traditional ARM approach does not consider the presence of taxonomies and it restricts the items in association rules to the leaf-level items in the taxonomy [Ghosh and Nath, 2004; Zaki, 2001]. It is valuable, however, to find rules across different levels of the taxonomy due to the following reasons:

1. Rules at lower levels may not have minimum support; thus many significant associations may not be discovered if we restrict rules to items at the leaves of the taxonomy
2. Taxonomies can be used to prune uninteresting or redundant rules.

Tsur et al. [1997] formulated Query Flocks, an algorithm for generalising the market-basket problem. They achieved this by parameterizing the query with a filter condition to eliminate values of the parameters that are uninteresting. The query flocks algorithm is a generalisation of the A-Priori technique.

Dehuri and Mall [2006] developed the multi-objective evolutionary algorithm called improved niched Pareto genetic algorithm (INPGA). It is an algorithm for mining highly predictive and comprehensible classification rules from large databases. INPGA strongly emphasizes predictive accuracy and comprehensibility of the rules.

In ARM, frequent pattern mining (FPM) is the most time-consuming part. The traditional ARM algorithms that extract all frequent patterns are not very efficient due to the fact that a large portion of the frequent patterns are redundant. This presents users with a lot of uninteresting or redundant rules. Association rules, however, can be extracted in a generalized form conveying knowledge in a compact manner with the use of taxonomies.

3. The Generalization of Association Rule Mining Algorithm (GARM)

The problem of discovering generalised association rules is handled in three steps:

1. Find all itemsets whose support is greater than the user-specified minimum support as proposed by Hipp et al. [1998].
2. Extract the rules from frequent itemsets
3. Find the generalized association rules by pruning the taxonomy using genetic algorithms. We apply comprehensibility, surprise, interestingness and confidence measures to form a coherent set of four complementary measures to extract interesting rules.

3.1. Frequent Itemset Generation

To generate the frequent itemset (FI), a lattice is formed that contains all the $2^{|L|}$ subsets of the itemset L. The FI's are generated by traversing the lattice in such a way that all frequent itemsets are found but as few infrequent ones as possible are visited. This is achieved by algorithms that use the downward closure property. The downward closure property states that: All subsets of frequent itemsets must also be frequent.

Using the support of a candidate itemset, a candidate may be pruned or added to the frequent itemsets. The strategy of the algorithm is to join every two frequent $(k-1)-$itemsets which have a $(k-2)-$prefix in common. Such a join results in a candidate $k-$itemset. After the support of a candidate is counted, it is pruned or added to the set of frequent itemsets depending on its value. This approach starts with the 1-itemsets as the set of candidate 1-itemsets. All frequent candidates are added for further candidate generation. This ensures that all frequent itemsets are visited and reduces the number of the visited infrequent itemsets.

3.2. Lattice Traversal

There are several mechanisms for traversing the tree (lattice traversal) including breadth first search (BFS) as used in Apriori based algorithms Agrawal [1993]. In BFS the frequent (k-1) itemsets are discovered when generating the candidate k-itemsets. These algorithms exhibit good performance because they prune the candidates that have infrequent subset before counting their supports. The depth-first search (DFS) lattice traversal based algorithms on the other hand are restricted to mine only frequent $k-$itemsets with $k > 3$. They also do not guarantee that the infrequent subsets of a candidate C are predetermined at the time the support of C has to be determined. This limits them in that candidates having infrequent subsets cannot be pruned hence causing operating overheads. Especially when mining generalized rules this problem becomes impeding because the pruning is required for optimization purposes.

3.3. Support Counting

Counting actual occurrences of candidates as done by Apriori relies on a hash tree structure Agrawal [1993]. Counting candidates that occur infrequently is not computationally demanding. But with growing candidate sizes, this approach causes performance degradation because the number of levels of the hash tree increases. In this work, support is counted by intersecting transaction identifiers is done in Partition and Eclat, Hipp *et. al.*[1998].

3.4. Prune the Taxonomy Using Genetic Algorithms

We use a genetic algorithm to prune the taxonomy of the generated frequent itemsets (FI). A set of three complementary metrics (confidence, comprehensibility and
J-Measure) is used as criteria for pruning the FIs.

4. Design of the Genetic Algorithm

In this section the design of the genetic algorithm for generalised rule induction is presented.

4.1. Knowledge Representation

A crucial issue in the design of an individual representation is to decide whether the candidate solution represented by an individual will be a rule set or just a single rule Freitas [2003]. Genetic algorithms (GAs) for rule discovery can be divided into two broad approaches, the Michigan approach and the Pittsburgh approach [Dehuri *et al*. 2006]. The biggest distinguishing feature between the two is that in the Michigan approach (also referred to as Learning Classifier Systems) an individual is a single rule, whereas in the Pittsburgh approach each individual represents an entire set of rules.

In the context of this research the use of the term Michigan approach will denote any approach where each GA individual encodes a single prediction rule. The choice between these two approaches strongly depends on which

kind of rule is to be discovered. This is related to which kind of data mining task being addressed. Suppose the task is classification. Then evaluate the quality of the rule set as a whole, rather than the quality of a single rule. In other words, the interaction among the rules is important. In this case, the Pittsburgh approach seems more natural [Frietas 2002].

On the other hand, the Michigan approach might be more natural in other kinds of data mining tasks. An example is a task where the goal is to find a small set of high-quality prediction rules, and each rule is often evaluated independently of other rules. The Pittsburgh approach directly takes into account rule interaction when computing the fitness function of an individual. However, this approach leads to syntactically-longer individuals, which tends to make fitness computation more computationally expensive. In addition, it may require some modifications to standard genetic operators to cope with relatively complex individuals.

By contrast, in the Michigan approach the individuals are simpler and syntactically shorter. This tends to reduce the time taken to compute the fitness function and to simplify the design of genetic operators. However, this advantage comes with a cost. First of all, since the fitness function evaluates the quality of each rule separately, now it is not easy to compute the quality of the rule set as a whole - i.e. taking rule interactions into account. Another problem is that, since we want to discover a set of rules, rather than a single rule, we cannot allow the GA population to converge to a single individual which is what usually happens in standard GAs. This introduces the need for some kind of niching method. The need for niching in the Michigan approach may be avoided by running the GA several times, each time discovering a different rule. The drawback of this approach is that it tends to be computationally expensive.

We have, therefore, used a modified Michigan encoding/decoding scheme which associates two bits to each attribute. If these two bits are 00 then the attribute next to these two bits appears in the antecedent part and if it is 11 then the attribute appears in the consequent part. And the other two combinations, 01 and 10 will indicate the absence of the attribute in either of these parts. So the rule ACF □ BE will look like 00A 11B 00C 01D 11E 00F. In this way we can handle variable length rules with more storage efficiency, adding only an overhead of $2k$ bits, where k is the number of attributes in the database. The decoding is performed as follows:

$$DV = mnv + (mxv - mnv) * \left(\sum (2^{i-1} * i^{th} bitvalue) / \left(2^n - 1 \right) \right) \quad (1.1)$$

where DV is the decoded value; $1 \leq i \leq n$ and n is the number of bits used for encoding; mnv and mxv are minimum and maximum values of the attribute; and bitvalue is the value of the bit in position i. For brevity, this encoding scheme will not deal with relational operators and as such the rules generated from this formula will not include relational operators.

Due to the fact that there may be a large number of attributes in the database, we propose to use multi-point crossover operator. There are some difficulties to use the standard multi-objective GAs for association rule mining problems. In case of rule mining problems, we need to store a set of better rules found from the database. Applying the standard genetic operations only, the final population may not contain some rules that are better and were generated at some intermediate generations. The better rules generated at intermediate stages should be kept. For this task, an external population is used. In this population no genetic operation is performed. It will simply contain only the non-dominated chromosomes of the previous generation. At the end of first generation, it will contain the non-dominated chromosomes of the first generation. After the next generation, it will contain those chromosomes, which are non-dominated among the current population as well as among the non-dominated solutions till the previous generation.

The scheme applied here for encoding/decoding the rules to/from binary chromosomes is that the different values of the attributes are encoded and the attribute names are not. For encoding a categorical valued attribute, the market basket encoding scheme is used. For a real valued attribute their binary representation can be used as the encoded value. The range of values of that attribute will control the number of bits used for it.

The archive size is fixed, i.e., whenever the number of non-dominated individuals is less than the predefined archive size, the archive is filled up by dominated individuals. Additionally, the clustering technique used does not loose boundary points.

4.2. Individual encoding

In this paper an individual corresponds to a single association rule of the form where is the rule antecedent, consisting of a conjunction of conditions on the values of the predicting attributes, and is the rule consequent, which is constituted by a conjunction of conditions of the values of the predicted attributes. We encode the rule antecedent and the rule consequent separately. Both the antecedent and the consequent are separately encoded as a variable-length list of rule conditions. The rule's conditions can contain both propositional logic and first-order logic, in which the pair of operands is compatible, as defined in the Attribute-Compatibility Table (ACT) for the data being mined [Freitas 2003].

The genetic material of an individual's rule antecedent is randomly generated when the initial population is created and is thereafter subject to the action of crossover and mutation. In contrast, the production of the genetic material of an individual's rule consequent is treated in a special way, due to its strategic importance in determining the quality of the rule represented by the individual. The basic idea is to delay the generation of the rule consequent until fitness-computation time, when the rule consequent is generated in such a way

that the rule's predictive accuracy is maximized. A more elaborated description of this idea is as follows.

First of all, crossover and mutation operators are applied only to an individual's rule antecedent. Once these genetic operators have been applied, the just-produced rule antecedent of an offspring individual is matched against all the tuples of the database, to compute statistics to be used for calculating the individual's fitness. These statistics contain the frequency of tuples satisfying the rule antecedent, distributed per goal attribute and per goal attribute value.

After computing the statistics to determine the rule's fitness, the rule's consequent part is generated by selecting the frequency that maximizes the predictive accuracy of the rule. The main reason for determining the consequent part after the antecedent part is that it maximizes the individual's predictive accuracy for a given rule antecedent; and it leads to a gain in computational efficiency, since by matching a single rule antecedent against the database just once we are computing, in a single database pass, the predictive accuracy associated with several possible rules. This is significantly more efficient than matching the entire rule (antecedent and consequent) against the database for each possible rule. Note also that this is important because predictive accuracy computation usually by far dominates the processing time of a GA mining a large database.

4.3. Fitness Assignment

The fitness function used in this paper consists of three metrics including. We combine these metrics into an objective fitness function. The complementary set of measures include confidence defined in equation (1), comprehensibility defined in equation (2) and J-Measure defined in equation (4).

The following expression is used to quantify the comprehensibility of an association rule

$$Comprehensiblity = \log(1+|Y|) + \log(1+|X \cup Y|) \quad (1.2)$$

where, $|Y|$ and $|X \bigcup Y|$ are the number of attributes involved in the consequent part and the total rule, respectively.

The confidence factor or predictive accuracy of the rule is given in the following equation

$$Confidence = \sigma(X \cup Y)/\sigma(X) \quad (1.3)$$

The J-Measure is a good indicator of the information content of the generated rules. In rule inference we are interested in the distribution of the rule "implication" variable Y, and especially its two events y and complement. The purpose is to measure the difference between the priori distribution f(y), i.e. f(Y = y) and f(Y $\neq$ y), and the posteriori distribution f(Y | X). The J-Measure gives the average mutual information between the events y and f(Y = y). The J-Measure shows how dissimilar our a priori and posteriori beliefs are about Y meaning that useful rules imply a high degree of dissimilarity. The J-Measure is calculated as:

$$JM = f(y|x)\log 2\left(\frac{f(y|x)}{f(y)}\right) + (1 - f(y|x)\log 2\left(\frac{(1 - f(y|x)}{1 - f(y)}\right) \quad (1.4)$$

where JM is the J-Measure.
The fitness function is calculated as the arithmetic weighted average confidence, comprehensibility and J-Measure. The fitness function ($f(x)$) is given by:

$$f(x) = \frac{W_1 * Comprehensibility + W_2 * (J - Measure) + W_3 * Confidence}{W_1 + W_2 + W_3} \quad (1.5)$$

where $W_1, W_2 W_3$ are user-defined weights.

4.4. Environmental Selection

The number of individuals contained in the archive is constant over time, and the truncation method prevents boundary solutions being removed. During environmental selection, the first step is to copy all non-dominated individuals, i.e., those which have a fitness lower than one, from archive and population to the archive of the next generation. If the non-dominated front fits exactly into the archive the environmental selection step is complete. In case the archive is too small, the best dominated individuals in the previous archive and population are copied to the new archive. Otherwise, truncate the archive.

5. Experiments

The experimental design we employed in this paper was to generate a table of frequent itemsets and then generate transactions by picking itemsets from I and inserting them into the transaction. The process starts with building taxonomy over the items. The taxonomy is created by assigning children to the roots, and then to the nodes at the next depth until the frequent itemsets are exhausted.

The data used were data from supermarket customer purchases. There are 25,000 items and the taxonomy has 3 levels with 55 roots. The total number of transactions in the database is 350,000. The number of generalised rules discovered in more than one and a half times more that in cumulate. This shows a good performance of the designed algorithm. We evaluated the performance of this algorithm and compared it with an implementation of Cumulate [10]. Default values of the parameters are: Population size = 40, Mutation rate = 0.5, Crossover rate = 0.8, Selection in Pareto Archive (elitism)= 0.5. The stopping criterion used is the non evolution of the archive during 10 generations, once the minimal number of generations has been over passed.

5.1 Results and Discussion

Minimum Support was varied from 2.5% to 0.5%. GARM was faster than Cumulate by approximately 1.5, with the performance gap increasing as the

minimum support decreased. The number of transactions was varied from 1,000 to 10,000. GARM performed better than Cumulate though they tend to converge as the number of transactions grows.

6. Conclusion And Future Works

In this paper we have combined a-priori query technique with a genetic algorithm to deal with generalisation of the association rule mining problem. The results show that our proposed model can attain considerable performance improvement in terms of minimum support and number of items. The intended future improvements include consideration of various other characteristics that a good generalisation algorithm should have. These characteristics include a limitless number of roots and/or levels in the taxonomy, depth-ratio, and number of transactions. We also hope subject the algorithm to larger sample sizes and different data types.

References

AGRAWAL R., IMIELINSKI T., SWAMI A., (1993). Mining Association Rules Between Sets of Items in Large Databases. Proc. of the 1993 ACM SIGMOD Conf. on Management of Data

AGRAWAL R., SRIKANT R.(1994). Fast Algorithms for Mining Association Rules. Proc. of the 20th Int'l Conf. on Very Large Data Bases

DEHURI S., JAGADEV K. , GHOSH A. AND MALL R. (2006). Multi-objective Genetic Algorithm for Association Rule Mining Using a Homogeneous Dedicated Cluster of Workstations. American Journal of Applied Sciences 3 (11): 2086 – 2095, 2006 ISSN 1546-9239, 2006 Science Publications

FONSECA M. C. AND FLEMING J. P. (1998) Multi – objective Optimization and Multiple Constraint Handling with Evolutionary Algorithms-Part I: A Unified Formulation. IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans, 28(1):26-37

FREITAS A. A., (2003) A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery. Advances in evolutionary computing: theory and applications, Pp 819 845

GHOSH A. AND NATH B.,(2004). Multi – objective rule mining using genetic algorithms. Information Sciences 163

pp 123- 133

HIPP J., MYKA A., WIRTH RU¨ DIGER, AND GUNTZER. (1998). A new Algorithm for Faster Mining of Generalised Association Rules. Proceeding of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98), pages 75 – 82, Nantes, France, September 1998. Springer.

KHABZAOUI M., DHAENES C., AND TALBI E. (2005). Parallel Genetic Algorithms for Multi-Objective rule mining. MIC2005. The 6th

Metahueristics International Conference, Vienna, Austria. SRIKANT R. AND AGRAWAL R. (1995). Mining Generalized Association Rules. Proceedings of the 21st VLDB Conference Zurich, Switzerland, 1995

TSUR D., ULLMAN J. D., ABITEBOUL S., CLIFTON C., MOTWAMI R., AND NESTOROV S. (1997). Query Flocks: A Generalization of Association Rule Mining.

ZAKI, M.J. (2001). Generating non-redundant association rules. In Proceedings of the Sixth ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY: ACM, 34-43.

ZHAO Q. AND BHOWMICK S. S. (2003). Association Rule Mining: A Survey. Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116 , 2003.