Extraction of Interesting Association Rules using GA Optimization

Virendra Kumar Shrivastava1 Dr. Parveen Kumar2 Dr. K. R. Pardasani3

Abstract—Association rule mining is a process of discovering interesting and unexpected rules form very large databases. Discovery of association rules at primitive-level is called single-level association rules or primitive-level association rules. However, mining association rules at multi-level may lead to the discovery of more specific and useful knowledge from dataset. Mining of Multi-level Association Rules (MLAR) are not useful until it can be used to improve decision making process. The main hurdle in this process is the number of rules grows exponentially with the number of items. Support and confidence limit the level of interestingness of the generated rules. However, the challenge arises in selection of interesting rules from the set of rules. In this research paper, we endeavor to optimize the rules generated by FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases using genetic algorithm. The rules may optimize using measures like support, confidence factor, interestingness and completeness.

Keywords—Discovery of multi-level association rules, interestingness, completeness and Genetic Algorithm.

I. INTRODUCTION

Techniques of association rule mining can be used to discover unknown or hidden correlation between items found in the database of transactions. An association rule [1,2,4,5,8,11,12] is a rule, which implies certain association relationships among a set of attributes (such as ‘occurs together’ or ‘one implies to other’) in a database. Apriori [5] is the most popular and influential algorithm to find all the frequent itemsets. It is proposed by Agrawal and Srikant in 1994. It is also called the level-wise algorithm. Multi-level association rules mining involves items at different level of abstraction. For many applications, it is difficult to find strong association among data items at low or primitive level of abstraction. Associations discovered at higher levels may represent common sense knowledge. Multi-level techniques find rules that are hidden or impossible to mine when searching at the primitive-level. This is because the conventional algorithms neglect several items from analysis that do not appear often enough to be considered significant. For example, joystick may not be purchased frequently, and therefore omitted from association rules. However, by using concept hierarchies, we can place joystick in a larger category containing mouse, pen drive, mouse pad, and joystick etc., called computer accessories. Thus, indirectly include their name in association mining process. To discover multilevel association rules, one need to provide (i) data at multi-levels of abstraction and (ii) efficient methods for multi-level rule mining. Researchers have given some methods for MLARM [5,7,10,15]. In this research, we considered FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases [3]. But all rules generated by this method may not be interesting. The main hurdle in this process is the number of rules grows exponentially with the number of items. The rules may optimize using measures like support, confidence factor, interestingness and completeness. The prime objective of this paper is to find interesting rules (of high predictive accuracy) from given data set using optimization of Genetic Algorithm.

This paper is organized as follows. The section two describes the basic concepts related to the multiple level association rules. In section three, discuss the Genetic Algorithm. Section four describes the results, and finally we conclude our research work in section five.

II. MULTIPLE-LEVEL ASSOCIATION RULES

The multi-level association rule mining utilizes a concept hierarchy. This hierarchy represents the relationship among different concept levels. For example, in figure 1, we represent a concept hierarchy that one might find in a typical electronics / computer sales house. To explore the mining of association rules from a largest set of transaction data, let assume that the database contains:

i. an item data set which contains the description of each item in I in the form of <Ai, description>, where Ai ∈ I, and

ii. a transaction data set T , which consists of a set of transactions <Ti, {Ap, . . .,Aq}>, where Ti is a transaction identifier and Ai ∈ I (for i ≠ p, . . ., q).

Definition: A pattern or an itemset A, is one item Ai or a set of conjunctive items Ai ^ ... ^ Aj, where Ai, . . . , Aj ∈ I. The support of a pattern A in a set S, s(A/S), is the number of transactions (in S) which contain A versus the total number of transactions in S. The confidence of A => B in S, c(A => B/S), is the ratio of s(A ^ B/S) versus s(A/S), i.e., the probability that pattern B occurs in S when pattern A occurs in S. To generate relatively frequent occurring patterns and reasonably strong rule implications, one may specify two thresholds: minimum support s’, and minimum confidence c’. Observe that, for finding multiple-level association rules,
different minimum support and/or minimum confidence can be specified at different levels.

Figure 1: concept hierarchy

We attempt to optimize the rules generated by FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases [3]. This rule generation method uses hierarchy information encoded in transaction table instead of the original transaction table. This is because; first a data mining query is usually in relevance to only a portion of the transaction database, such as computer, printer etc instead of all the items. Thus, it is useful to first collect the relevant set of data and then work repeatedly on the task related set. Second, encoding can be performed during the collection of task related data and then there is no extra encoding pass required. Third, an encoded string, which represents a position in a hierarchy, requires lesser bits than the corresponding bar code. Thus, it is often beneficial to use an encoded table. For example, the item 'IBM Desktop Computer' is encoded as '111' in which the first character, '1', represents 'Computer' at level-1, the second, '1', for 'laptop (computer)' at level-2, and the third, '1', for the brand 'IBM' at level-3. Repeated items (i.e., items with the same encoding) at any level will be treated as one item in one transaction.

The FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases consists of two main phases. Phase one is the construction of a modified Frequent Pattern tree. Phase two is the repetitive building of small data structures, the actual mining for these data structures, and their release. The association rules are generated at multiple- level using the frequent patterns at related concept level.

III. GENETIC ALGORITHM

Genetic algorithm (GA) was first developed by John Holland at University of Michigan in 1975. It incorporates Darwinian evolution theory with sexual reproduction. GA is a stochastic search algorithm modeled on the process of natural selection, which underlines biological evolution. GA has been successfully applied in many search, optimization, and machine learning problems. A group of individuals called population, is stored and modified during each iteration of the algorithm. In GA’s iterations are referred to as generations. GA processes generations by generating new populations of strings from old ones. Every string is the encoded binary, real etc., version of a candidate solution. An evaluation function associates a fitness measure to every string indicating its fitness for the problem. Standard GA[9,13] apply genetic operators such selection, crossover and mutation on an initially random population in order to compute a whole generation of new strings. It generates solution for successive generations. The probability of an individual reproducing is proportional to the goodness of the solution it represents. Therefore, the quality of solutions in successive generations improves. The GA process is terminated when an acceptable or optimum solutions is found.

The function of GA is as follows:

• **Selection** deals with the probabilistic survival of the fittest, in that more fit chromosomes are chosen to survive. Where fitness is a comparable measure of how well a chromosome solves the problem at hand.

• **Crossover** specifies how the genetic algorithm combines two individuals, or parents, to form a crossover child for the next generation.

• **Mutation** alters the new solutions so as to add stochasticity in the search for better solutions. This is the chance that a bit within a chromosome will be flipped (0 becomes 1 and vice versa).

**Fitness function:** Ideally the discovered rules should have (a) a high predictive accuracy; (b) be comprehensible; and (c) be interesting. The accomplishment of a genetic algorithm is directly linked to the accuracy of the fitness function. The fitness function should be customized to the specific search spaces. We take a fitness function that considers major issues in evaluating an individual against its search space. The fitness function should be customized to the specific search spaces. We take a fitness function that considers major issues in evaluating an individual against its search space. The fitness of a population is the sum of the individual fitness values of that population. The fitness function is the primary performance sink in a genetic algorithm, because this is the place that the underlying data must be accessed. Therefore, optimization should be considered wherever possible. The general structure of a rule is defined as:

IV. **IF ANTECEDENT THEN CONSEQUENT**

Let a rule be of the form:

IF A THEN C,

Where A is the antecedent (a conjunction of conditions) and C is the consequent (predicted class). The predictive performance of a rule can be summarized by a $2 \times 2$ matrix, sometimes called a confusion matrix, as depicted in the following figure 2:

![Confusion Matrix](image)

Figure 2: Confusion Matrix for a rule

The abbreviation and meaning of the labels used in the confusion matrix have the following meaning:

- **TP** = True Positives = Number of examples satisfying A item set and item set C
- **FP** = False Positives = Number of examples not satisfying item set A but satisfying item set C
FN = False Negatives = Number of examples satisfying item set A but not satisfying item set C
TN = True Negatives = Number of examples not satisfying A nor C

It is clear that the higher the values of TP and TN, and the lower the values of FP and FN, the better the rule.

Interestingness Factor (INF) = TP/(TP+FP)

Now measure the predictive accuracy of a rule by taking into account not only its INF but also a measure of how “complete” the rule is, i.e. what is the proportion of examples having the predicted class C that is actually covered by the rule antecedent. The rule completeness factor measure, denoted CF, is computed as:
Completeness Factor (CF) = TP / (TP+FN)

In order to combine the INF and CF measures one can define a fitness function such as:
Fitness = INF x CF

Although this fitness function does a good job in evaluating predictive performance, it has nothing to say about the comprehensibility of the rule. This fitness function can be extended (or any other focusing only on the predictive accuracy of the rule) with a rule comprehensibility measure in several ways. A simple approach is to define a fitness function such as:
Fitness = w1 × (INF × CF) + w2 × S

Where, S is a measure of rule simplicity. The S values lie between [0, 1] and w1 and w2 are user-defined weights. In general, its value is inversely proportional to the number of conditions in the rule antecedent – i.e., the shorter the rule, the simpler it is.

V. OPTIMIZATION METHODOLOGY

We applied GA over the rules generated from FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases [3]. Optimization [14,16] does not mean maximization or minimization. Optimization is means to get the most feasible solution or utilization of the available methodology for their best uses. The following genetic algorithm is used to optimize (i.e. finding interesting relationships) rules at level l.

1. Create random population of n chromosomes.
2. Calculate fitness for each chromosome in the population
3. Selection – based on fitness function
4. Apply Cross-over and mutation on the selected members
5. Accept or reject new one
6. Replace old with new population
7. Test problem criterion
8. Repeat step 2-7 until criterion is satisfied

The genetic algorithm mechanism can be explained with the following flow chart (as given in figure 3).

VI. RESULTS

We have applied GA to the rules obtained by the FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules. We have implemented genetic algorithm for optimization in MATLAB [15]. We have used computer user’s feedback dataset to test the effectiveness of proposed algorithm. We have used MATLAB 7.0 and tested all experiments on a Dell Laptop with Intel® Core™2 Duo 2.0 GHz processor and 2.00 GB of main memory using Microsoft Windows XP operating system. The following table shows the values of TP, FP, FN, interestingness, completeness and fitness of the rule.

<table>
<thead>
<tr>
<th>TP</th>
<th>FP</th>
<th>FN</th>
<th>INF</th>
<th>CF</th>
<th>Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>12</td>
<td>7</td>
<td>0.625</td>
<td>0.741</td>
<td>0.0463</td>
</tr>
<tr>
<td>34</td>
<td>23</td>
<td>16</td>
<td>0.596</td>
<td>0.68</td>
<td>0.405</td>
</tr>
<tr>
<td>35</td>
<td>30</td>
<td>14</td>
<td>0.538</td>
<td>0.714</td>
<td>0.384</td>
</tr>
<tr>
<td>8</td>
<td>32</td>
<td>14</td>
<td>0.2</td>
<td>0.364</td>
<td>0.072</td>
</tr>
<tr>
<td>25</td>
<td>34</td>
<td>12</td>
<td>0.424</td>
<td>0.676</td>
<td>0.286</td>
</tr>
<tr>
<td>16</td>
<td>29</td>
<td>16</td>
<td>0.356</td>
<td>0.5</td>
<td>0.177</td>
</tr>
</tbody>
</table>

Table 1: obtained values of TP, FP, FN, interestingness, completeness and fitness function.

Given below, Table 2 describes parameters used in the genetic algorithm implementation

<table>
<thead>
<tr>
<th>Selection</th>
<th>Tournament, size = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crossover probability</td>
<td>0.1</td>
</tr>
<tr>
<td>MutationProbability</td>
<td>0.005</td>
</tr>
<tr>
<td>Fitness function</td>
<td>Discussed in section 3.</td>
</tr>
<tr>
<td>GA population</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 2: Genetic algorithm parameters

Our experimental results show that the optimized rules have a high interestingness and completeness.

VII. CONCLUSION

In this research work, we have used multi-level association rules generated by FP-tree and COFI Based Approach. All
generated rules are not interesting. We have applied Generic Algorithm to optimize the association rules. We obtain a fitness function for the task of optimization and find the optimum solutions that are interesting rules. It extracts interesting rules with predictive accuracy.

VIII. REFERENCES


2) R Srikant, Qouc Vu and R Agrawal. “Mining Association Rules with Item Constrains”. IBM Research Centre, San Jose, CA 95120, USA.


7) Jiawei Han and Yongjian Fu “Discovery of Multiple-Level Association Rules from Large Databases”. Proceedings of the 21st VLDB Conference Zurich, Switzerland, 1995.


10) Jiawei Han and Yongjian Fu “Discovery of Multiple-Level Association Rules from Large Databases”. IEEE Trans. on Knowledge and Data Eng. Vol. 11 No. 5 pp 798-804, 1999.

11) J. Han, J, Pei and Y Yin. “Mining Frequent Patterns Without Candidate Generation”. In ACM SIGMOD Conf. Management of Data, May 2000.


Global Journals Guidelines Handbook 2010

www.GlobalJournals.org