International Journal of Computer Applications (0975 - 8887)
Volume 76 - No. 16, August 2013

GARM: A Simple Graph Based Algorithm For Association Rule Mining

Amal Dev P
Department of Computer Science and Engineering
Government Engineering College Painavu
Idukki, Kerala, India

Sobhana N V
Department of Computer Science and Engineering
Rajeev Gandhi Institute of Technology
Kottayam, Kerala, India

Philumon Joseph
Department of Computer Science and Engineering
Government Engineering College Painavu
Idukki, Kerala, India

ABSTRACT

Association rule mining is an important component of data mining. In the last years a great number of algorithms have been proposed with the objective of solving the obstacles presented in the generation of association rules. In this work, a new graph based algorithm for associative rule mining which has so many advantages over the existing methods is proposed. It can be used to improve decision making in a wide variety of applications such as: market basket analysis, medical diagnosis, bio-medical literature, protein sequences, census data, logistic regression, fraud detection in web, CRM of credit card business etc.

Keywords:

Apriori, Weighted Graph, Association Rule Mining

1. INTRODUCTION

Association rule mining, one of the most important and well researched techniques of data mining, was first introduced in [1]. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories. Association rules are widely used in various areas such as telecommunication networks, market and risk management, inventory control etc. Various association mining techniques and algorithms will be briefly introduced and compared later.
The main application areas of association rule mining includes market basket analysis, medical diagnosis, protein sequences, census data, CRM of credit-card diagnosis, was explained in [2]. It is all about finding some kind of patterns or relationships between various datasets.
Traditionally, association analysis was considered an unsupervised technique, so it has been applied in knowledge discovery tasks. Recent studies have shown that knowledge discovery algorithms, such as association rule mining, can be successfully used for prediction in classification problems. In these cases, the algorithm used for generating association rules must be tailored to the particularities of the prediction in order to build more effective classifiers. The improvement of association rules algorithms is the subject of many works in the literature. The main drawbacks of existing association rule mining algorithms are:

(1) Large number of database scans.
(2) Low algorithm performance.
(3) Obtaining non interesting rules.

In this work, a new method for associative rule mining is introduced. The proposed system works by constructing a graph. The main advantage is that the system needs only a single database scan, it is also possible to mine rules related to some particular items only.
The paper is organized as follows. Section 2 mentions about the existing work. Section 3 describes the proposed system. In section 5 the advantages of the proposed system is discussed. Finally, the conclusion is presented in section 5.

2. EXISTING SYSTEMS

The main methods that are existing for association rule mining are apriori algorithm, Eclat algorithm [3], FP-growth algorithm [2,6], GUHA procedure [7], ASSOC [4] and OPUS [8,9] Search algorithm. The also exists so many other algorithms which are modified version of these base algorithms.
The main algorithm apriori [1] uses a ”bottom up” approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. Apriori uses breadth-first search and a Hash tree structure to count candidate item sets efficiently. It generates candidate item sets of length from item sets of length . Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent -length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

3. PROPOSED SYSTEM

Formally associative rule mining [5] can be defined as : Let I = $I_1; I_2; ...; I_m$ be a set of binary attributes, called items. Let T be a database of transactions. Each transaction t is represented as a binary vector, with t[k] = 1 if t bought the item I_k, and t[k] = 0 otherwise. There is one tuple in the database for each transaction. Let X be a set of some items in I. A transaction t is said to be satisfies X if for all items I_k in X, t[k] = 1.
Rakesh Agrawal et. al.[5] defined associative rules as: An association rule means an implication of the form $X \implies I_j$, where X is a set of some items in I, and I_j is a single item in I that is

not present in X. The rule $X \implies Ij$ is satisfied in the set of transactions T with the confidence factor $0 \le c \le 1$ iff at least c% of transactions in T that satisfy X also satisfy I_j. A notation $X => I_j|c$ is used to specify that the rule $X \implies I_j$ has a confidence factor of c, where confidence is a measure of the rule's strength, support corresponds to statistical significance
The proposed system works in two steps: In step 1 a graph is being constructed from the given database of transactions. This is the only step on which a database scan is needed. In step 2 association rules are derived from the graph that is resulted from step 1. For the easiness of operations a new graph which is named as GARM graph has been introduced.
The GARM graph is similar to the common graph G=(V,E) where V is the set of vertices and E is the set of edges. The difference is that in GARM graph V is a tuple (Vertex V, colours); where the set "colours" denoted by colours represent the colours of edges that are connected to the vertex represented as "V", and E is a triple (Vertex1, Vertex2, colour C) which means that the vertices "Vertex1" and "Vertex2" are connected through an edge coloured as "C". In other words GARM is a multigraph with coloured edges. The main operations that are interested are insertion, coloured sub-graph extraction (i.e. if a colour is given, the algorithm will return the graph, whose edges with specified colour). The process is explained in the following subsections:

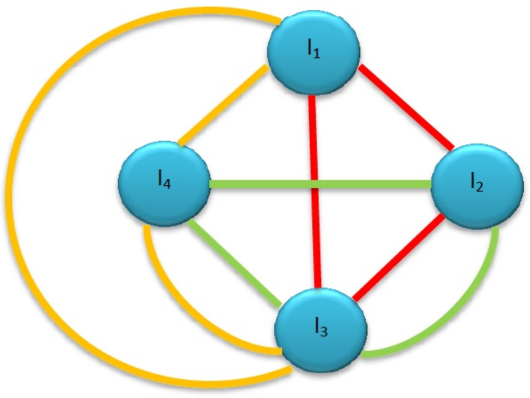

Fig. 1. GARM graph for T

3.1 Phase 1: Graph Construction phase

Algorithm 1 GARM Construction Algorithm

```
 1: procedure CREATE_GRAPH($T$)
 2:     $color = 0$
 3:     for each transaction $t$ in $T$ do
 4:         for each item $I_i$ in $t = I_1, I_2, ..., I_n$ do
 5:             if $I_i \notin V$ then
 6:                 $V = VUI_i$
 7:             end if
 8:         end for
 9:         for all $I_j \in t \neq I_i$ do
10:             $E = EU(I_i, I_j, color)$
11:         end for
12:         $V(I_i, colour) = V(I_i, colourUcolor)$
13:         $color = color + 1$
14:     end for
15: end procedure
```

Algorithm 2 Coloured Subgraph Construction Algorithm

```
 1: procedure FIND_SUBGRAPH($V$)
 2:     Vertex set $V_{new} = \emptyset$
 3:     Edge set $E_{new} = \emptyset$
 4:     for each vertex $ver$ in $V$ do
 5:         if $\exists$ $c$ in $\{color\}$ of $ver$ | $c \in \{color\}$ of $V$ then
 6:             $V_{new} = V_{new}Uver$
 7:             if (V, ver) $\in$ E then
 8:                 $weight[(V, ver)] = weight[(V, ver)] + 1$
 9:             else
10:                 $E_{new} = E_{new}U(V, ver, color)$
11:                 $weight[(V, ver)] = 1$
12:             end if
13:         end if
14:     end for
15:     return graph G=($V_{new}, E_{new}$)
16: end procedure
```

The graph construction can be explained with the help of the algorithm 1.The algorithm works as follows: Consider each itemset in transaction as a vertex in GARM graph. Firstly, For each transactional item-set $I_1, I_2, ..., I_n$ construct a complete graph of size 'n' and assign a previously unassigned colour to the edges of this graph. The next step is to merge this graph with the GARM graph, i.e. an union operation is done between edge set of newly created graph and edge set of GARM graph.
An example of function **CREATE_GRAPH**() with input database of transactions T as:

$$T_1 = I_1I_2I_3$$
$$T_2 = I_2I_3I_4$$
$$T_3 = I_1I_3I_4$$

can be illustrated in figure 1

3.2 Phase 2: Rule mining phase

Next step is to derive association rules from GARM graph. Before going into that function, let us present an operation that to find the coloured subgraph from GARM, which is connected to a specific vertex "V". The pseudocode for procedure is given in algorithm 2.

The algorithm 2 works in the logic that if two vertices have same coloured edge connected to it, then it means that the both vertices are connected. Step 7-11 is used to reduce GARM graph to well known weighted graph. **FIND_SUBGRAPH**() can also be done by finding and merging the coloured spanning tree of all colours to which the specified vertex is connected.
Now the association rule mining step. The algorithm for the process is as in algorithm 3 which can be explained as: The procedure first of all remove all edges from the resultant subgraph which have the weight less than the specified minimum support count (min_sup). This step will help us to remove irrelevant associations. Then check for the paths from each vertices, to create rules related to that item.

Algorithm 3 Algorithm for Rule mining

```
1: procedure MINE_RULE($min\_sup, conf$)
2:     for each vertex 'v' in G do
3:         G1 = FIND_SUBGRAPH(v)
4:         $E_{new} = E_{new} - E(v,u) \mid w[(v,u)] < min\_sup$
5:         if w[$(v,u_1)$] = w[$(v,u_2)$] = ... = w[$(v,u_n)$] =$Max\_Weight(G) \mid u_1 \neq u_2 \neq ... \neq u_n$ then
6:             R = R U $\{v \longrightarrow u_1, u_2, ..., u_n\}$
7:         else if w[$(v,u_1)$] = w[$(u_1,u_2)$] = ... = w[$(u_{n-1},u_n)$] = $Max\_Weight(G) \mid u_1 \neq u_2 \neq ... \neq u_n$ then
8:             R = R U $\{v \longrightarrow u_1, u_2, ..., u_n\}$
9:         else if w[$(v,u_1)$] = w[$(u_1,u_2)$] = ...= w[$(u_{i-1},u_i)$] $\neq$ w[$(u_i,u_{i+1})$]= ... = w[$(u_{n-1},u_n)$] $\mid u_1 \neq u_2 \neq ... \neq u_n$ then
10:            R = R U $\{vu_1u_2...u_i \longrightarrow u_{i+1}, u_{i+2}, ..., u_n\}$
11:        else w[$(v,u_1)$] = w[$(u_1,u_2)$] = ...= w[$(u_{i-1},u_i)$] $\neq$ w[$(u_i,u_{i+1})$]$\neq$ ... $\neq$ w[$(u_{n-1},u_n)$]$\mid$ $u_1 \neq u_2 \neq ... \neq u_n$
12:            R = R U $\{vu_1u_2...u_i \longrightarrow u_{i+1}, u_{i+2}, ..., u_n\}$ U $\{vu_1u_2...u_{i+1} \longrightarrow u_{i+2}, ..., u_n\}$ U ... U $\{vu_1u_2...u_i...u_{n-1} \longrightarrow u_n\}$
13:        end if
14:    end for
15: end procedure
```

The calling of procedure **CREATE_GRAPH**() will result in the creation of GARM graph. Then the **MINE_RULE**() procedure will mine rules by creating coloured subgraph of all vertices.

3.3 Complexity

The complexity of proposed algorithms is as follows: Suppose the dataset of transaction contains 'n' item-sets and totally 'm' items in entire dataset and the longest item-set contains 'k' items. Then the algorithm 1 take $O(nk)$ time where 'n' to take complete transactions and 'k' for creating edges between interrelated items which is denoted as vertices. Algorithm 2 take $O(km)$ time where 'k' is for taking all the colours that are connected to the vertex, 'm' for checking all vertices for the colour. Finally algorithm 3 take $O(km^2)$ where, 'km' for procedure **SUB_GRAPH** and 'm' for generating rules related to all 'm' vertices.

3.4 Memory Management

This graph based algorithm for association rule mining effectively utilises the memory constraints. The only memory that is needed here is to store the GARM graph. GARM graph can be efficiently implemented by storing the vertices, which was represented as a structure containing vertex id and set of colours. The set of colours can be represented in two ways: One way, is to store all the id's w.r.to the colours that the vertex contains and another way is to store each vertex as a bit-vector in which each '1' in i'th position means i'th coloured edge is connected to the vertex.

4. ADVANTAGES OF PROPOSED SYSTEM

There are several advantages for the proposed system over the existing system. The major advantages are:

4.1 Number of database scan

The apriori algorithm needs a large number of database scans and generate large candidate sets. But the database scans are too costly and time consuming, So the total cost for running apriori algorithm is too high. Whereas, the GARM algorithm read the database only at the time of GARM graph construction. That is there is only one database scan.

4.2 Dynamic updation

The proposed system supports dynamic updation. If the admin wants to add a new transaction after creating the GARM graph, then:

—Create a graph for new transaction.
—Add newly created graph with existing GARM graph.

Suppose an error get identified in some transaction after GARM creation, then the following steps can be done to remove that invalid transaction:

—Create a graph for the erroneous transaction.
—Subtract similar graph from GARM graph.

Since these operations needs unit time, the dynamic updations are quite easy and consistent.

4.3 Mining rules related to interested items only

The existing algorithms result in a set of associative rules. But suppose that someone is interested only associative rule that contains some particular item I_i then the existing algorithm fails. Solution for this is doing the steps 3 to 12 of algorithm 3 with respect to vertex I_i. This is the main advantage of the proposed algorithm.

4.4 Simple operations

The algorithms that was discussed earlier can be denoted and accomplished with simple operations. That is, the algorithms can be done in the following ways:

—Algorithm 1: Initialize vertex corresponding to I_i as $\emptyset$ and for each occurrence of I_i union the set of I_i with the colour of new graph.
—Algorithm 2: Add vertex I_j to coloured subgraph of I_i, if $I_i \bigcap I_j \neq \emptyset$. Otherwise this algorithm can be implemented by creating and merging coloured spanning tree.

These arithmetic set operations can be done in simple and efficient manner with less computation time.

4.5 Less complexity

The proposed algorithm is of less complexity, both in the terms of time and space when compared to the existing systems. Because, int his algorithm candidate sets are not generated and no time consuming operations are done in this algorithm. Whereas, the naive procedure require setting up of 2^m counters corresponding to all subsets of the set of items I, where m is number of items in I [11].

5. CONCLUSION

In this paper a new graph based algorithm for association rule mining is proposed. The GARM will override the existing solutions with high efficiency and less time complexity. The main strategy of this paper is by decomposing the main problem into two sub problems. The idea is mainly bagged with finding GARM graph from the given set of transactions and mining rules from the GARM graph.The advantages of proposed system is as mentioned in section IV. The option for item specific rule mining can be said as the main advantage of the proposed system.

Acknowledgement

The authors would like to thank **Prof. Sreeram S.** and **Prof. Jiju P.**, Dept. of CSE, MEA Engg. College, Kerala for their insightful support and advices.

6. REFERENCES

[1] Agrawal, Rakesh; and Srikant, Ramakrishnan; Fast algorithms for mining association rules in large databases, in Bocca, Jorge B.; Jarke, Matthias; and Zaniolo, Carlo; editors, Proceedings of the 20th International Conference on Very Large Data Bases (VLDB), Santiago, Chile, September 1994, pages 487-499

[2] Witten, Frank, Hall: Data mining practical machine learning tools and techniques, 3rd edition

[3] Zaki, M. J. (2000). ”Scalable algorithms for association mining”. IEEE Transactions on Knowledge and Data Engineering 12 (3): 372??390. doi:10.1109/69.846291

[4] Hjek, Petr; and Havrnek, Tom (1978). Mechanizing Hypothesis Formation: Mathematical Foundations for a General Theory. Springer-Verlag. ISBN 3-540-08738-9

[5] Webb, Geoffrey I. (2000); Efficient Search for Association Rules, in Ramakrishnan, Raghu; and Stolfo, Sal; eds.; Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2000), Boston, MA, New York, NY: The Association for Computing Machinery, pp. 99-107

[6] Data Mining Concepts and Techniques; Jiawei Han and Micheline Kamber; Second Edition

[7] Raunch, Jan; Logical calculi for knowledge discovery in databases; Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery 4(2):217-240

[8] Webb, Geoffrey I. (1995); OPUS: An Efficient Admissible Algorithm for Unordered Search, Journal of Artificial Intelligence Research 3, Menlo Park, CA;

[9] Bayardo, Roberto J., Jr.; Agarwal, Rakesh; Gunopulos, Dimitrios(2000). ”Constraint-based rule mining in large, dense databases”. Data mining and knowledge discovery 4 (2):217-240

[10] Rakesh Agrawal, Tomasz Imielinski and Arun Swami; ”Mining Association Rules between Sets of Items in Large Databases”Proceedings of the 1993 ACM SIGMOD Conference