A DISTRIBUTED ALGORITHM FOR MINING FUZZY ASSOCIATION RULES

George Stephanides
University of Macedonia, Department of Applied Informatics
156 Egnatia Street, 540 06 Thessaloniki GREECE

Mihai Gabroveanu, Mirel Cosulschi, Nicolae Constantinescu
University of Craiova, Computer Science Department
13 A.I. Cuza Street, 200585 Craiova ROMANIA

Keywords: data mining, fuzzy association rules, distributed mining.

Abstract: Data mining, also known as knowledge discovery in databases, is the process of discovery potentially useful, hidden knowledge or relations among data from large databases. An important topic in data mining research is concerned with the discovery of association rules. The majority of databases are distributed nowadays.

In this paper is presented an algorithm for mining fuzzy association rules from these distributed databases. This algorithm is inspired from DMA (Distributed Mining of Association rules) algorithm for mining boolean association rules.

1 INTRODUCTION

Data mining, also known as knowledge discovery in databases, is the process of discovery potentially useful, hidden knowledge or relations among data from large databases. An important task in data mining process is the discovery of association rules. The majority of databases are distributed nowadays.

The task of discovering association rules was first introduced in (Agrawal R., 1993). Many of proposed algorithms for mining association rules are sequential algorithms. The most popular are: Apriori (Rakesh Agrawal, 1994), DHP, DIC. The basic problem of finding fuzzy association rules was introduced in (Chan Man Kuok, 1998).

Mining association rules based on fuzzy sets can handle quantitative and categorical data, providing the necessary support to use uncertain data types with existing algorithms. Today the majority of databases are distributed. The records of transactions corresponding to each customer operation registered in a stores chain distributed in many locations form an example of such databases. The main problem here is to discover the association rules from these distributed data.

In this paper we introduce an algorithm for mining fuzzy association rules from these distributed databases. This algorithm is an adaptation of DMA algorithm used here for mining fuzzy association rules.

2 PROBLEM DEFINITION

2.1 Sequential problem definition

The formal problem definition as in (Chan Man Kuok, 1998) is the following:

Let \( DB = \{t_1, \ldots, t_n\} \) a transactional database. We consider this database is characterized by a set of categorical or quantitative attributes (items). Let \( I = \{i_1, \ldots, i_m\} \) the set of these attributes. We note with \( \text{dom}(i_k) \) the domain of values for the attribute \( i_k \). For each attribute \( i_k \), \((k = 1, \ldots, m)\) we will consider \( n(k) \) associated fuzzy sets. Let \( F_{i_k} = \{f_{i_1}^k, \ldots, f_{i_m}^k\} \) be the set of fuzzy sets. For an attribute \( i_k \) and a fuzzy set \( f_{i_k}^k \), the membership function is \( \mu_{f_{i_k}^k} \).

Definition 2.1. We call fuzzy itemset the tuple \( \langle X, F_X \rangle \), where \( X \subseteq I \) and \( F_X \) is a set of fuzzy sets associated with items from \( X \). A fuzzy itemset \( \langle X, F_X \rangle \) is called \( k \)-fuzzy itemset if the number of attributes from \( X \) is \( k \).

Definition 2.2. A fuzzy association rule is an implication with following form \( X \in A \Rightarrow Y \in B \), where \( X, Y \in I \), \( X \cap Y = \emptyset \), \( X = \{x_1, \ldots, x_p\} \), \( Y = \{y_1, \ldots, y_q\} \), \( A = \{a_1, \ldots, a_p\} \) and \( B = \{b_1, \ldots, b_q\} \) are fuzzy sets related to attributes from \( X \), respectively \( Y \). More exactly, \( a_i \in F_{x_i}, (i = 1, \ldots, p) \), and \( b_i \in F_{y_i}, (i = 1, \ldots, q) \).
We denote this rule with \((X, A) \Rightarrow (Y, B)\).

The intuitively signification of this fuzzy association rule \((X, A) \Rightarrow (Y, B)\) is: “if a transaction (tuple) satisfies the property \(X \in A\) then it will satisfy the property \(Y \in B\) with a high probability also”.

**Definition 2.3.** The fuzzy support value of itemset \((X, F_X)\) in DB is:

\[
FS_{(X, F_X)} = \frac{\sum_{i \in DB} \prod_{j \in X} \alpha_{a_j}(t_i[x_j])}{|DB|}
\]

where

\[
\alpha_{a_j}(t_i[x_j]) = \begin{cases} 
\mu_{a_j}(t_i[x_j]), & \text{if } \mu_{a_j}(t_i[x_j]) \geq \omega \\
0, & \text{otherwise}
\end{cases}
\]

and \(\omega\) is a user specified minimum threshold for the membership function. Thus, the values of membership functions lesser than this minimum threshold are ignored.

**Definition 2.4.** A fuzzy itemset \((X, F_X)\) is called a large (frequent) fuzzy itemset if its fuzzy support value is greater than or equal to the minimum support threshold (minsup), namely \(FS_{(X, F_X)} \geq \text{minsup}\).

An association rule is considered as interesting if it has enough support and high confidence value. This association rule can be encountered under the name strong rule.

**Problem 1** (Sequential Mining Fuzzy Association Rules). Given the database DB characterized by a set of attributes \(I\), the fuzzy sets associated with attributes from \(I\), \(\omega\) the minimum support threshold for membership function, the minimum support threshold (minsup) and the minimum confidence threshold (minconf), extract all interesting fuzzy association rules.

**Definition 2.5.** Let \((X, A) \Rightarrow (Y, B)\) be a fuzzy association rule. The fuzzy support value of the rule is defined as fuzzy support value of the itemset \((X, Y, \{A, B\})\):

\[
FS_{(X, A) \Rightarrow (Y, B)} = FS_{(X,Y),\{A,B\}}
\]

**Definition 2.6.** A fuzzy association rule is called a frequent rule if its fuzzy support value is greater than or equal to the minimum support threshold (minsup), namely \(FS_{(X, A) \Rightarrow (Y, B)} \geq \text{minsup}\).

Based on discovered large fuzzy itemsets we can generate all possible frequent rules, but in order to be interesting they must have a high confidence value.

**Definition 2.7.** Let \((X, A) \Rightarrow (Y, B)\) a fuzzy association rule. The fuzzy confidence value of the rule is defined as:

\[
FC_{(X, A) \Rightarrow (Y, B)} = \frac{FS_{(Z, C)}}{FS_{(X, A)}}
\]

where \(Z = \{X, Y\}\) and \(C = \{A, B\}\)

The confidence of the rule is defined as the fraction between the value of fuzzy support of the fuzzy itemset \((Z, C)\) and the value of fuzzy support of the fuzzy itemset \((X, A)\).

**Lemma 1.** If a fuzzy itemset \((X, F_X)\) is a large fuzzy itemset in DB, \(Y \subseteq X, F_Y \subseteq F_X\), then also fuzzy itemsets \((Y, F_Y)\) are large in DB.

From the above lemma we can draw the conclusion that any fuzzy subitemset of a large fuzzy itemset is also large.

The problem of sequential mining of fuzzy association rules can be decomposed in two subproblems:

1. find all large fuzzy itemsets.
2. generate the fuzzy association rules from the large fuzzy itemsets founded.

The majority of algorithms for mining fuzzy association rules (see (Gyenesi, 2000), (Hong T.P., 2000)) are based on the algorithm Apriori (Rakesh Agrawal, 1994).

### 2.2 Distributed problem definition

Let \(DB = \{DB_1, DB_2, \ldots, DB_n\}\) be a distributed database over \(n\) sites \(S_1, S_2, \ldots, S_n\). We denote with \(D\) the number of transactions from DB, and with \(D_i\) the number of transactions from \(DB_i\), for all \(i = 1, \ldots, n\).

**Definition 2.8.** For a given fuzzy itemset \((X, F_X)\) we call global fuzzy support value the fuzzy support value of \((X, F_X)\) in DB defined as:

\[
FS_{(X, F_X)} = \frac{\sum_{i \in DB} \prod_{j \in X} \alpha_{a_j}(t_i[x_j])}{|DB|}
\]

and global fuzzy support count in DB is defined as:

\[
CFS_{(X, F_X)} = \sum_{i \in DB} \prod_{j \in X} \alpha_{a_j}(t_i[x_j])
\]

**Definition 2.9.** For a given fuzzy itemset \((X, F_X)\) and a database \(DB_i\) we call local fuzzy support value in \(DB_i\) the fuzzy support value of \((X, F_X)\) in \(DB_i\) defined as:

\[
FS_{(X, F_X)}^{i} = \frac{\sum_{t_i \in DB_i} \prod_{j \in X} \alpha_{a_j}(t_i[x_j])}{|DB_i|}
\]

and local fuzzy support count in \(DB_i\) is defined as:

\[
CFS_{(X, F_X)}^{i} = \sum_{t_i \in DB_i} \prod_{j \in X} \alpha_{a_j}(t_i[x_j])
\]

Let \(\text{minsup}\) be the minimum support threshold.

**Definition 2.10.** A fuzzy itemset \((X, F_X)\) is called global large fuzzy itemset if \(FS_{(X, F_X)} \geq \text{minsup}\).
Definition 2.11. A fuzzy itemset \((X, F_X)\) is called local large fuzzy itemset at site \(S_i\) if \(FS_i^{(X, F_X)} \geq \minsup\).

Definition 2.12. If a fuzzy itemset \((X, F_X)\) is both globally large and locally large at a site \(S_i\), it is called gl-large fuzzy itemset at site \(S_i\).

In the following, we will denote with \(L\) the set of all globally large fuzzy itemsets in \(DB\), and with \(L_{(k)}\) the set of all globally large \(k\)-fuzzy itemsets in \(DB\).

Problem 2 (Distributed Mining Fuzzy Association Rules). Given the set of items \(I\), the distributed database \(DB = \{DB_1, DB_2, \ldots, DB_n\}\), the fuzzy sets associated with attributes from \(I\), the minimum support threshold (\(\minsup\)) and the minimum confidence threshold (\(\minconf\)), extract all global fuzzy association rules.

3 THE DISTRIBUTED ALGORITHM

In (Cheung D.W., 1996), the authors proposed a DMA algorithm for mining boolean association rules from distributed databases.

3.1 Generate set of candidate fuzzy itemsets

The candidate fuzzy itemsets reduction is made on the basis of the properties of the global large fuzzy itemsets and local large fuzzy itemsets subsequently presented:

Lemma 2. If a fuzzy itemset \((X, F_X)\) is locally large at a site \(S_i\), then all its subsets are also locally large at site \(S_i\).

Lemma 3. If a fuzzy itemset \((X, F_X)\) is globally large, then there exist a site \(S_i\), \((1 \leq i \leq n)\), such that \((X, F_X)\) is locally large at site \(S_i\).

Lemma 4. If a fuzzy itemset \((X, F_X)\) is gl-large fuzzy itemset at a site \(S_i\), \((1 \leq i \leq n)\), then all its sub-fuzzy itemsets, \((Y, F_Y)\), \(Y \subseteq X\), are also gl-large fuzzy itemsets at site \(S_i\).

We use \(GL_i\) to denote the set of all gl-large fuzzy itemsets at site \(S_i\), and \(GL_{(k)}\) to denote all \(k\)-gl-large fuzzy itemsets at site \(S_i\).

Lemma 5. If \((X, F_X) \in L_{(k)}\), (i.e. is a globally large fuzzy \(k\)-itemset), then there exists a site \(S_i\), \((1 \leq i \leq n)\) such that \((X, F_X)\) and all its \((k-1)\) sub-fuzzy itemsets are gl-large fuzzy itemsets at site \(S_i\).

Like in the DMA algorithm, which is an adaptation of the Apriori algorithm, at \(k\)-th iteration, the set of candidate sets is obtained by applying the FuzzyApriori.Gen function on \(L_{(k-1)}\). We denote this set by \(CA_{(k)}\). More exactly,

\[ CA_{(k)} = \text{FuzzyApriori.Gen}(L_{(k-1)}) \]

For each site \(S_i\), \((1 \leq i \leq n)\), we denote with \(CG_{(k)}^{i}\) the set of candidate fuzzy itemsets generated applying FuzzyApriori.Gen on \(GL_{(k-1)}^i\), i.e.,

\[ CG_{(k)}^i = \text{FuzzyApriori.Gen}(GL_{(k-1)}^i) \]

Because \(GL_{(k-1)}^i \subseteq L_{(k-1)}\), then \(CG_{(k)}^i\) is a subset of \(CA_{(k)}\). Following, we denote \(CG_{(k)} = \bigcup_{i=1}^{n} CG_{(k)}^i\).

Theorem 1. For every \(k > 1\), the set of all globally large \(k\)-fuzzy itemsets \(L_{(k)}\) is a subset of \(CG_{(k)} = \bigcup_{i=1}^{n} CG_{(k)}^i\).

Applying the Theorem 1 the result is that we can use the set \(CG_{(k)}\), which is a superset of \(L_{(k)}\), as a candidate set instead of \(CA_{(k)}\), and could be much smaller that \(CA_{(k)}\).

Thus the candidate set for \(L_{(k)}\) will be generated at \(k\)-th iteration in the following manner: first the set of candidate sets \(CG_{(k)}\) can be generated locally at each site \(S_i\). After this step, sites exchange fuzzy support count and compute the set of gl-large fuzzy itemsets \(GL_{(k)}\). Based on \(GL_{(k)}\), the candidate fuzzy itemsets at \(S_i\) for \((k+1)\)-st iteration can then be generated.

3.2 Local pruning of candidate sets

The Lemma 3 can be used to perform a local pruning of the set of candidate fuzzy item sets. At a site \(S_i\), after the set of candidate fuzzy itemsets \(CG_{(k)}^i\) is generated, in order to find if a candidate fuzzy itemset \((X, F_X) \in CG_{(k)}^i\) is gl-large fuzzy itemset, the fuzzy support count must be requested from all other sites. We can prune this request for fuzzy support count for some candidates using a local pruning technique. The basic idea is that at site \(S_i\), if a candidate fuzzy itemset \((X, F_X) \in CG_{(k)}^i\) is not locally large at site \(S_i\), there is no need for \(S_i\) to compute global support to find out if it is globally large. This is possible because in this case, either \((X, F_X)\) is not globally large, or it will be locally large at some other site, and hence only the sites where \((X, F_X)\) is locally large need to be responsible to find its global support count. We use \(LL_{(k)}^i\) to denote those fuzzy candidate item sets in \(CG_{(k)}^i\) which are locally large at site \(S_i\).

3.3 The algorithm outline

In Algorithm 1 is presented in detail the FUZZY-DMA algorithm for distributed mining of association rules.
A DISTRIBUTED ALGORITHM FOR MINING FUZZY ASSOCIATION RULES

Algorithm 1 FUZZY-DMA

INPUT:
- $DB_1, \ldots, DB_n$ - the database partition at each site.
- $\text{minsup}$ - the minimum support threshold.
- $\mathcal{F}$ - the set of fuzzy sets associated with attributes from $T$.

OUTPUT:
- $L$ - the set of all globally large fuzzy itemsets in $DB$.

METHOD: For all $k \geq 1$, iterates the following algorithm distributively at each site $S_i$. At the end of each step a synchronization is required to develop global count. The algorithm terminates when either $L(k)$ returned is empty or candidate $CG(k)$ is empty.

1: if $k = 1$ then
2: $T_1^{(1)} = \text{Get Local Fuzzy Count}(DB_i, \emptyset, 1)$
3: else
4: $CG(k) = \bigcup_{i=1}^n CG(k_i) = \bigcup_{i=1}^n \text{Fuzzy Apriori Gen}(GL_{(k-1)}(i))$
5: $T(k) = \text{Get Local Fuzzy Count}(DB_i, CG(k), i)$
6: for all $(X, A) \in CG(k)$ do
7: if $CFS_i(X, A) \geq \text{minsup} \times D_i$ then
8: insert $(X, A)$ into $LL_i^{(k)}$ {Broadcast support count request to compute global fuzzy support count}
9: for $j = 1, \ldots, n; j \neq i$ do
10: broadcast Count Request($LL_i^{(k)}$, $S_j$) {Receive support count request}
11: for $j = 1, \ldots, n; j \neq i$ do
12: receive $LL_j^{(k)}$ extract $CFS_j(X, A)$ from $T_j^{(k)}$ and send to $S_i$ {Compute global fuzzy support count}
13: for all $(X, A) \in LL_i^{(k)}$ do
14: receive $CFS_j(X, A)$ from sites $S_j, where j \neq i$
15: $CFS(X, A) = \sum_{p=1}^n CFS_p$
16: if $CFS(X, A) \geq \text{minsup} \times D$ then
17: insert $(X, A)$ into $G_i(k)$
18: broadcast $G_i(k)$ {Compute global $L(k)$}
19: receive $G_k(i)$ from all other sites $S_j, (i \neq j)$
20: $L(k) = \bigcup_{i=1}^n G_i(k)$
21: return $L(k)$

to compute the local fuzzy support count $CFS_i(X, A)$ and store it into the hash tree $T_i^{(k)}$ using function Get Local Fuzzy Count, and generate set of locally large fuzzy itemsets $LL_i^{(k)}$. After this, $S_i$ broadcasts the candidate fuzzy itemsets from $LL_i^{(k)}$ to other sites to collect fuzzy support counts. The fuzzy support counts are needed to compute global support counts and generate set of all gl-large $k$-fuzzy itemsets at site $S_i$.

Finally computed gl-large fuzzy itemsets are broadcasted to all other sites, and these can compute $L(k)$.

The algorithm is stopped when either $L(k)$ returned is empty or candidate set $CG(k)$ is empty.

4 CONCLUSION

In this article, it is proposed an algorithm for mining fuzzy association rules from distributed databases more efficiently than a sequential algorithm. In the future, we will study the means of automatically finding fuzzy sets associated with database attributes. The other direction of improvement is related to the study of new relationships between local and global large itemsets in order to reduce the number of messages exchanged among sites.

REFERENCES


Cheung D.W., Jiawei Han, N. V. F. A. Y. F. (1996). A fast distributed algorithm for mining association rules. In In 4th International Conference on Parallel and Distributed Information Systems (PDIS '96), pages 31–43. IEEE Computer Society Technical Committee on Data Engineering, and ACM SIGMOD.

