Mining for Mutually Exclusive Items in Transaction Databases

George Tzanis and Christos Berberidis

Department of Informatics, Aristotle University of Thessaloniki

Thessaloniki 54124, Greece

{gtzanis, berber, vlahavas}@csd.auth.gr

URL : http://mlkd.csd.auth.gr

ABSTRACT

Association rule mining is a popular task that involves the discovery of co-occurences of items in transaction databases. Several extensions of the traditional association rule mining model have been proposed so far, however, the problem of mining for mutually exclusive items has not been directly tackled yet. Such information could be useful in various cases (e.g. when the expression of a gene excludes the expression of another) or it can be used as a serious hint in order to reveal inherent taxonomical information. In this paper, we address the problem of mining pairs of items, such that the presence of one excludes the other. First, we provide a concise review of the literature, then we define this problem, we propose a probability-based evaluation metric, and finally a mining algorithm that we test on transaction data.

INTRODUCTION

Association rules are expressions that describe a subset of a transaction database. When mining for such patterns it is quite often that we come up with a large number of rules that appear to be too specific and not very interesting. A rule that relates two specific products in a market basket database it is not very likely to be really strong compared to a rule that relates two groups or two families of products. Hierarchical relationships among items in a database can be used in order to aggregate the weak, lower level rules into strong, higher level rules, producing *hierarchical, multiple level* or *generalised association rules*. However, such information is not always explicitly provided although it might exist.

Mining for taxonomies is a really challenging task that, to the best of our knowledge, has not been approached yet. Taxonomies are conceptual hierarchies, implemented by *is-a* relationships. The discovery of such relationships would involve the complete description and formulation of concepts that are more general or more specific than others. To learn taxonomies from data implies the automatic extraction of human concepts from the data, with the use of an algorithm. In our understanding this is virtually impossible. However, we believe that when mining for various types of patterns, one can get serious hints about possible hierarchical relationships. Let us say for instance that a supermarket customer is vegetarian. Then it would be really rare for this customer to buy both veggie burgers *and* red meat. It seems that the two products *exclude* each other. When one of them is present then the probability to also find the other one is very low. Motivated by that observation we propose a method for mining for *mutually exclusive* items. Such information is also useful regardless its use as a taxonomy clue. In this paper we define the problem of mining for mutually exclusive items. We propose a probability-based mutual exclusion metric and a mining algorithm that we test on transaction data.

The paper is organized as follows. The next section presents the required background knowledge. Section 3 contains a short review of the relative literature. Section 4 contains the description of the proposed approach, definitions of terms and notions used, the proposed algorithm, a novel metric for measuring the mutual exclusion and an illustrative example of our approach. In section 5 we present our experiments and in section 6 we discuss the presented approach. Finally, section 7 contains our conclusions and our ideas for future research.

PRELIMINARIES

The association rules mining paradigm involves searching for co-occurrences of items in transaction databases. Such a co-occurrence may imply a relationship among the items it associates. These relationships can be further analyzed and may reveal temporal or causal relationships, behaviors etc.

The formal statement of the problem of mining association rules can be found in (Agrawal et al., 1996). Given a finite multiset of transactions *D*, the problem of mining association rules is to generate all association rules that have support and confidence at least equal to the user-specified minimum support threshold (*min_sup*) and minimum confidence threshold (*min_conf*) respectively.

The problem of discovering all the association rules can be decomposed into two subproblems (Agrawal et al., 1993):

1. The discovery of all itemsets that have support at least equal to the user-specified minimum support threshold. These itemsets are called *large* or *frequent* itemsets.
2. The generation of all rules from the discovered frequent itemsets. For every frequent itemset *F*, all non-empty subsets of *F* are found. For every such subset *S*, a rule of the form $S \Rightarrow F\text{-}S$ is generated, if the confidence of the rule is at least equal to the minimum user-specified confidence threshold.

Another method to extract strong rules is the use of *concept hierarchies*, also called *taxonomies* that exist in various application domains, such as market basket analysis. A taxonomy is a concept tree, where the edges represent "is-a" relationships from the child to the parent. Example of such a relationship is: "*Cheddar is-a cheese is-a dairy product is-a food is-a product*". When a taxonomy about a domain of application is available, a number of usually high-confidence rules that are too specific (having low support) can be merged, creating a rule that aggregates the support and therefore the information, in a higher abstraction level, of the individual rules. In other words, "looser" associations at the lower levels of the taxonomy are summarized, producing "winner" associations of higher levels. For example, the rule "*if a customer buys 0.5 lb. wheat bread then he/she also buys 1 lb. skimmed milk*" is very likely to have low support, while a rule "*if a customer buys bread then he/she also buys milk*" is very possible that it has much higher support, because it includes all types, brands and packages of bread and milk bought by the customers of the store.

RELATED WORK

Association rules were first introduced by Agrawal et al. (1993) as a market basket analysis tool. Later, Agrawal & Srikant (1994) proposed Apriori, a level-wise algorithm, which works by generating candidate itemsets and testing if they are frequent by scanning the database. Apriori exploits the downward closure property, according to which any non-empty subset of a frequent itemset is also frequent. Therefore, at each level the candidate frequent itemsets are generated based only on the frequent itemsets found in the previous level. About the same time Mannila et al. (1994) discovered independently the same property and proposed a variation of Apriori, the OCD algorithm. A joint paper combining the previous two works was later published (Agrawal et al., 1996). Several algorithms have been proposed since then, others improving the efficiency, such as FPGrowth (Han et al., 2000) and others addressing different problems from various application domains, such as spatial (Koperski & Han, 1995), temporal (Chen & Petrounias, 2000) and intertransactional rules (Tung et al., 2003), which can be also used for prediction (Berberidis et al., 2004).

One of the major problems in association rules mining is the large number of often uninteresting rules extracted. Srikant & Agrawal (1995) presented the problem of mining for generalized association rules. These rules utilize item taxonomies in order to discover more interesting rules. For example, given that "*gouda is-a cheese*" and "*cheddar is-a cheese*", we discover a rule such as "*if a customer buys bread then he/she also buys cheese*", with support higher than of a rule for a specific cheese. The authors propose a basic algorithm as well as three more efficient algorithms, along with a new interestingness measure for rules, which uses information in the taxonomy. Thomas & Sarawagi (1998) propose a technique for mining generalized association rules based on SQL queries. Han & Fu (1995) also describe the problem of mining "multiple-level" association rules, based on taxonomies and propose a set of top-down progressive deepening algorithms. Teng (2002) proposes a type of augmented association rules, using negative information called dissociations. A dissociation is relationship of the form "*X does not imply Y*", but it could be that "*when X appears together with Z, this implies Y*". Han & Fu (1994) propose some algorithms for dynamic refinement and dynamic generation of concept hierarchies. The generation of concept hierarchies concerns only numerical attributes and is based on data distribution. The dynamic refinement of a given or even a generated concept hierarchy is based on a particular learning request, the relevant set of data and database statistics.

Another kind of association rules are negative association rules. Savasere et al. (1998) introduced the problem of mining for negative associations. Negative associations deal with the problem of finding rules that imply what items are not likely to be purchased when a certain set of items is purchased. The approach of Savasere et al. demands the existence of a taxonomy and is based on the assumption that items belonging to the same parent of taxonomy are expected to have similar types of associations with other items. Since they consider only those cases where the expected support can be calculated based on the taxonomy, only a subset of the whole set of negative association rules can be discovered. They propose a naive and an improved algorithm for mining negative association rules along with a new measure of interestingness. In a more recent work Wu et al. (2004) present an efficient method for mining positive and negative associations and propose a pruning strategy and an

interestingness measure. Their method extends the traditional positive association rules ($A \Rightarrow B$) to include negative association rules of the form $A \Rightarrow -B$, $-A \Rightarrow B$, and $-A \Rightarrow -B$ (the minus sign means not). The last three rules indicate negative associations between itemsets *A* and *B*. A mutual exclusion can not be expressed by one such rule. If items *a* and *b* are mutually exclusive, then $\{a\} \Rightarrow -\{b\}$ and $\{b\} \Rightarrow -\{a\}$ concurrently, that is different from $-\{a\} \Rightarrow -\{b\}$.

MINING FOR MUTUALLY EXCLUSIVE ITEMS

In this section we describe our approach for discovering mutually exclusive items. Specifically, in subsection 4.1 we attempt to define the concept of mutual exclusion and we describe the intuition behind our approach. In subsection 4.2 we present the task of mining for contiguous frequent itemsets, which is a basic part of our approach. In section 4.3 we describe the metrics used for evaluating candidate pairs of mutually exclusive items and the algorithm for mining them. Finally, in subsection 4.4 we provide an illustrative example of our approach.

Problem Definition

Definition 1. Let *D* be a finite multiset of transactions and *I* be a finite set of items. Each transaction $T \in D$ is a set of items such that $T \subseteq I$. If two items $i_1 \in I$ and $i_2 \in I$ are *mutually exclusive*, then there is not any transaction $T \in D$, such that $\{i_1, i_2\} \subseteq T$.

The above definition of mutual exclusion is strict. However, the inverse of definition 1 does not generally stand, so it cannot be used to identify mutually exclusive items. Let’s think of the database of a large store containing over 10000 items and millions of transactions. It is possible that there is a large number of pairs of items that are never purchased together. According to the inverse of definition 1 all of these pairs of items are mutually exclusive. But, in fact only a very small number of them are truly mutually exclusive.

Based on the notions explained so far, we propose an algorithm for mining for mutually exclusive items in a transaction database and we focus on its application on market basket data. The *mutual exclusion* of two database items could imply an alternative type of association. Of course, this kind of mined knowledge needs to be confirmed by the domain expert so that it can be utilized eventually. A point that has also to be stressed here is although we perform a sort of causality analysis (the existence of an item excludes or "causes the non existence of" another), this is not what we aim for. Our goal is to discover items that never or rarely appear together under certain conditions that are explained below.

Mining for mutually exclusive items in a database possibly containing several thousands of different items, involves searching in a space that consists of all the possible pairs of items, because virtually any of them could contain two items that exclude each other. However this approach is naive and simplistic and can lead to many mutually exclusive items that in fact are not as explained above. We propose a more intuitive approach, which is based on the assumption/observation that every frequent itemset expresses a certain behavior of a group of customers and therefore it could be used to guide our search. Items that appear with high frequency in the subspace of a frequent itemset are more likely to be systematically mutually exclusive, because they follow a customer behavioral pattern and not because of pure chance or unusual cases. However, we cover these cases with an extra pass over the database in order to remove the candidate mutually exclusive pairs that for any reason are not confirmed by the transactions.

Our approach consists of three steps (Figure 1). In the first step, all the frequent itemsets are mined. Then, the frequent itemsets are used for mining the contiguous frequent itemsets, producing the extensions that will be used in the next step as candidate mutually exclusive items. Any frequent itemset mining algorithm can be used in the first step. Step 2 works in a level-wise manner like Apriori and requires a number of scans over the database, which is proportional to the size of the extensions discovered. In the case of the basic Apriori algorithm the number of scans is equal to the size of the itemsets, but there are some improvements in later versions that require less scans and they

are more efficient. The extensions of the contiguous frequent itemsets mined at the second step are candidates for participating in a pair of mutually exclusive items.

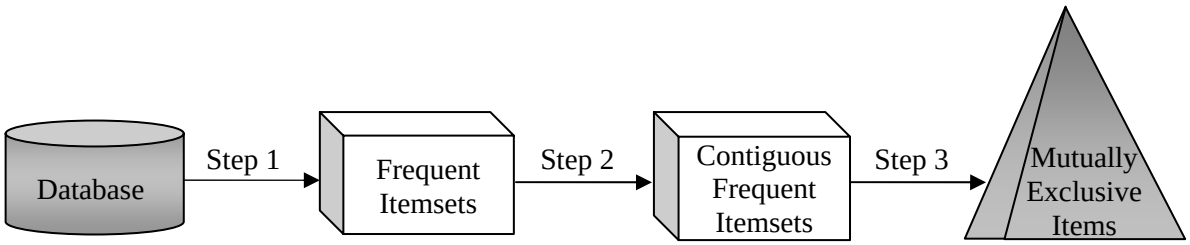

Figure 1: The three steps of our approach

Mining for Contiguous Frequent Itemsets

In the following lines we provide some definitions and formulate the problem of mining contiguous frequent itemsets as defined in our previous work Berberidis et al. (2005). Let D be a finite multiset of transactions and I be a finite set of items. Each transaction $T \in D$ is a set of items such that $T \subseteq I$. Mining for frequent k-itemsets involves searching in a search space, which consists of all the possible combinations of length k of all items in I. Every frequent itemset $F \subseteq I$ divides the search space in two disjoint subspaces: the first consists of the transactions that contain F and from now on will be called the *F-subspace* and the second all the other transactions.

Definition 2. Let $F \subseteq I$ be a frequent itemset in D, according to a first-level support threshold and $E \subseteq I$ be another itemset. The itemset $F \cup E$ is considered to be a *contiguous frequent itemset*, if $F \cap E = \varnothing$ and E is frequent in the F-subspace, according to a second level support threshold.

Itemset E is called the locally frequent extension of F. The term locally is used, because E may not be frequent in the whole set of transactions. In order to avoid any confusion, from now on we will use the terms local and locally, when we refer to a subset of D and the terms global and globally when we refer to D. For example, we call global support (*gsup*) the first-level support and local support (*lsup*) the second-level support. An itemset F that satisfies the minimum global support threshold

(*min_gsup*) is considered to be globally frequent and an itemset *E* that is frequent in the *F*-subspace, according to the minimum local support threshold (*min_lsup*), is considered to be locally frequent. The local support of an itemset *E* in the *F*-subspace can be calculated as in equation (1).

$$\mathrm{lsup}(E,F) = \frac{\mathrm{gsup}(E \cup F)}{\mathrm{gsup}(F)} \quad (\mathbf{1})$$

The local support threshold can be set arbitrarily by the user-expert or can be the same as the global support threshold. The contiguous frequent itemsets that contain a locally frequent extension of length *k* are called *k-contiguous frequent itemsets*.

Given a finite multiset of transactions *D*, the problem of mining contiguous frequent itemsets is to generate all itemsets $F \cup E$ that consist of an itemset *F* that has global support at least equal to the user-specified minimum global support threshold and an extension *E* that has local support at least equal to the user-specified minimum local support threshold.

Mutual Exclusion Metrics and Mining Algorithm

In order to distinguish when two items are mutually exclusive, we need a measure to evaluate the degree of the mutual exclusion between them. Initially, we should be able to evaluate this within the subspace of a frequent itemset (locally) and then it should be evaluated globally, with all the frequent itemsets that support this candidate pair to contribute accordingly. For this purpose, we propose the use of a metric we call MEM (Mutual Exclusion Metric) that can be calculated in two phases, the first one is local and is required for the second one, which is the global one.

Local Metric. We propose the following local metric (2), which will be called Local MEM for the evaluation of a candidate pair of mutually exclusive items that is supported by a frequent itemset I and its range is [0, 1].

$$LM_I(A,B) = \left[P(A-B) + P(B-A)\right] \min\left[P(A-B \mid A), P(B-A \mid B)\right] = \quad \textbf{(2)}$$
$$\left[(S_A - S_{AB}) + (S_B - S_{AB})\right] \min\left[\frac{(S_A - S_{AB})}{S_A}, \frac{(S_B - S_{AB})}{S_B}\right] =$$
$$(S_A + S_B - 2S_{AB})\left[1 - \frac{S_{AB}}{\min(S_A, S_B)}\right]$$

For the above formula $P(I) = 1$. S_X is the fraction of transactions that contain X over the number of transactions that contain I. Figure 2 shows that the Local MEM increases linearly when the local support of A and B increase, as long as there is no overlapping ($S_A, S_B < 0.5$ and $S_{AB}=0$), until it reaches its maximum value, which is 1.We assume that $S_A=S_B$ for simplicity, without any loss of generality. When the overlapping starts ($S_A, S_B > 0.5$), it drops fast down to zero, because even a small degree of overlapping is strong evidence against the possibility of two items to be mutually exclusive.

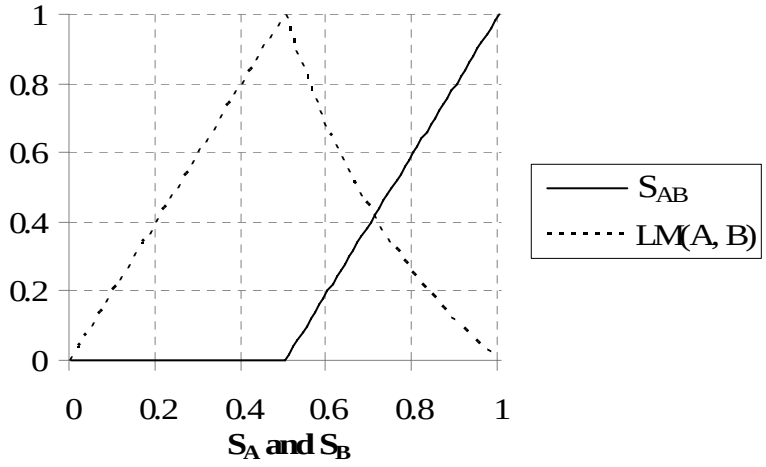

Figure 2: The Local MEM against the local support of the candidate mutually exclusive items

An illustrative example is depicted in Figure 3. In diagrams (a) and (b) the shadowed areas represent the transactions that support *A* but not *B* (or *A*-*B*) and *B* but not *A* (or *B*-*A*), which is the first factor in the Local MEM formula. The 2nd factor in the Local MEM equation is represented by the minimum of $\frac{A-B}{A}$ and $\frac{B-A}{B}$. The two items in the diagram (b) are less likely to be mutually exclusive than in (a).

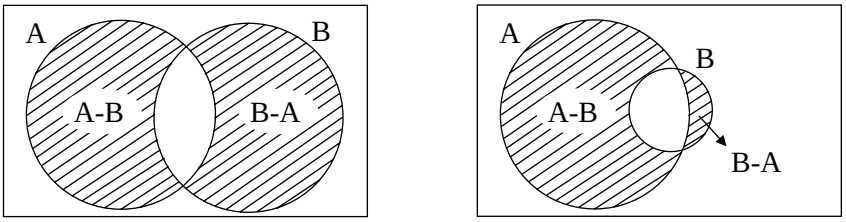

Figure 3: Venn diagrams

Global Metric. We propose the following global metric (3) for the evaluation of a candidate pair of mutually exclusive items that is supported by a set IS of frequent itemsets.

$$GM_1(A,B) = IIF\left(\sum_{I\in \mathrm{IS}} S_I LM_I(A,B)\right) \quad \textbf{(3)}$$

IIF stands for *Itemset Independence Factor* and is calculated as the ratio of the number of the distinct items contained in all itemsets that support a candidate pair over the total number of items contained in these itemsets. For example, the *IIF* of the itemsets {*A, B, C*} and {*A, D*} is 0.8, since there are 4 distinct items (*A, B, C* and *D*) over a total of 5 ones (*A, B, C, A* and *D*). The *IIF* is used in order to take into account the possible overlapping of two candidate mutually exclusive items. We do this, because the overlapping between the transactions that contain two different itemsets implies overlapping between the transactions that contain the pair. Alternatively, one can use (4), which is a normalized version of (3). The range of values of GM_1 is [0, +∞), while the range of GM_2 is [0, 1].

$$GM_2(A,B) = IIF\,\frac{\sum_{I\in IS} S_I LM_I(A,B)}{\max_{IS\in FS}\left(\sum_{I\in IS} S_I\right)} \quad \textbf{(4)}$$

FS is the set that contains all the sets of frequent itemsets that support a candidate pair of mutually exclusive items. The difference between (3) and (4) is the denominator in (4). This term is used in order to normalize the metric, since the numerator is always less or equal to the denominator. In other words, for each candidate pair we calculate the sum of the supports of the frequent itemsets that support the pair. The greatest sum is used to normalize the metric. This normalization is useful to the user who wants to know a priori the maximum value of the metric in order to use the proper threshold.

However, this is not desirable when the user wants to compare the results of different datasets or the results obtained by running the algorithm with different parameters. The algorithm for mining mutually exclusive pairs of items is presented in Table 1. For the first step, namely the discovery of frequent itemsets every frequent itemset mining algorithm can be used.

Table 1: The mutuallty exclusive items mining algorithm

Input: A multiset of transactions *D*, a minimum global support threshold *min_gSup*, a minimum local support threshold *min_lSup* and a minimum global MEM threshold *min_gMEM*.
Output: All mutually exclusive pairs.

```
FI ← mineFrequentItemsets(D, min_gSup)
for each (T ∈ D)
   for each (I ∈ FI)
      if (I ⊆ T)
         for each (E ∈ T-I)
            Extensions(I) ← Extensions(I) ∪ E
            Count[I][E] ← Count[I][E] + 1
for each (I ∈ FI)
   for each (E ∈ Extensions(I))
      if (Count[I][E] < min_lSup)
      Extensions(I) ← Extensions(I) - E
for each (T ∈ D)
   for each (I ∈ FI)
      for each (E1, E2 ∈ Extensions(I))
         if ({E1, E2} ⊆ T)
            ExtensionPairs(I) ← ExtensionPairs(I) ∪ {E1, E2}
            Local_MEM[I][{E1, E2}] ← calculateLocal_MEM(I, {E1, E2})
for each (I ∈ FI)
   for each (EP ∈ ExtensionPairs(I))
         AllExtensionPairs ← AllExtensionPairs ∪ EP
for each (EP ∈ AllExtensionPairs)
   if (Global_MEM(EP) ≥ min_gMEM)
      MutuallyExclusivePairs ← MutuallyExclusivePairs ∪ EP
return MutuallyExclusivePairs
```

Example

The following tables show an illustrative example. Table 2 contains a market basket dataset example and Table 3 the discovered association rules. The minimum support was set to 2/9 in order to extract these rules.

Table 2: A market basket dataset example

TID	Items in the Basket
1	espresso, sugar, newspaper
2	espresso, sugar, cola
3	espresso, sugar
4	cappuccino, cigarettes
5	cappuccino, sugar
6	cappuccino, sugar, sweets
7	decaf, sugar, chewing_gums
8	decaf, soda, vinegar
9	decaf, sugar, cigarettes

Table 3: Association rules mined from the dataset of Table 2

Association Rules	Support	Confidence
espresso $\Rightarrow$ sugar	3/9	1
decaf $\Rightarrow$ sugar	2/9	2/3
cappuccino $\Rightarrow$ sugar	2/9	2/3

After we applied our algorithm, we discovered the mutually exclusive pairs of items shown in Table 4, along with their metrics and corresponding frequent itemsets.

Table 4: Mutually exclusive items and their corresponding frequent itemsets mined from dataset of Table 2

Mutually Exclusive Pairs (Global MEM, Global Support)	**Frequent Itemsets**	
	Support	*Local MEM*
{espresso, cappuccino}: 5/9, 0	{sugar}: 7/9	0.714
{espresso, decaf}: 5/9, 0	{sugar}: 7/9	0.714
{cappuccino, decaf}: 4/9, 0	{sugar}: 7/9	0.571

EXPERIMENTS

In order to evaluate the performance of our algorithm we conducted a number of experiments on an IBM-Artificial market basket dataset (T10I4D100K). This dataset contains 100000 transactions. The graph in Figure 4 illustrates the performance of our mutually exclusive items mining algorithm in means of run time (seconds) while the minimum local support threshold varies from 0.1 to 0.3 and the minimum global support threshold varies from 0.01 to 0.04. As expected we observe that while the minimum local support threshold and the minimum global support threshold decrease, the run time of the algorithm increases. The performance degrades significantly when the two thresholds are set very low.

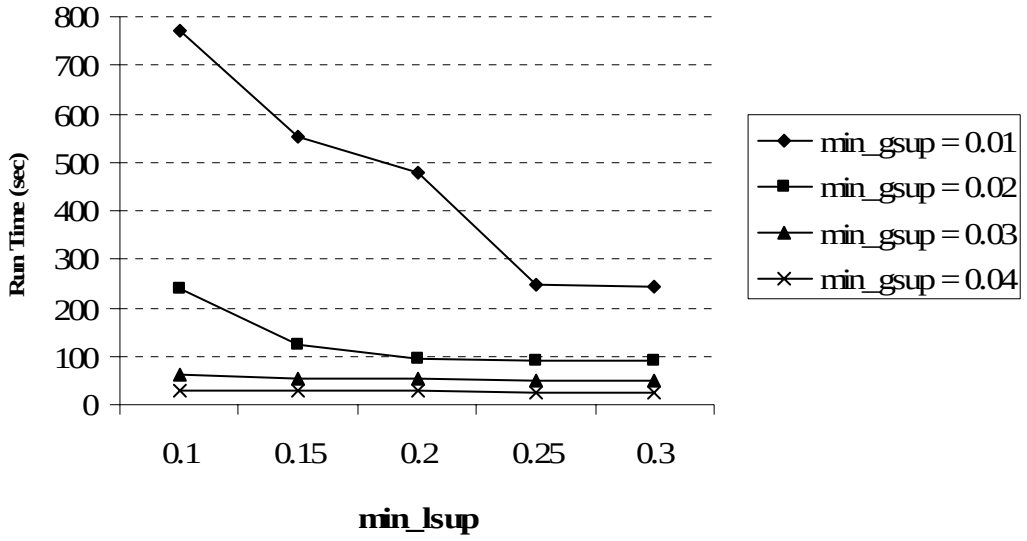

Figure 4: Run time of the mutually exclusive items mining algorithm depending on local support for four fixed vales of global support

DISCUSSION

We apply a level-wise technique in order to extract the contiguous frequent itemsets and then the mutually exclusive items. The intuition behind this approach is twofold: First, if these extensions are frequent indeed in the subspace of a frequent itemset then they could be important information about these itemsets, lost by a number of reasons. Second, if a large number of itemsets share the same extensions and these common extensions are frequent in the subspace of these itemsets, they are likely to be mutually exclusive and possibly of the same category and the same level of taxonomy. In such cases, the total support of the parent node in the taxonomy is broken down to many lower level supports, which are not high enough to satisfy the minimum set threshold and which explains the possible loss of potentially valuable knowledge. The support of the current itemset is reduced because of the low support of the extensions and eventually fails to qualify as a frequent itemset. When no taxonomy information is available in advance, the information gathered from this process can be a valuable hint about the taxonomy effect explained here and eventually the existence of a taxonomy.

Mining a transaction database to discover the underlying taxonomy is an elusive task with questionable results. In the existing applications, the existence of a taxonomy assists the miner to discover valuable knowledge. In our setup, the Holy Grail would be exactly the inverse procedure. In a mining problem where no taxonomy information is provided in advance, is it possible to follow a kind of reverse engineering procedure in order to mine nuggets of taxonomy information from the data? If so, then the benefit would be twofold. First, the domain expert would be provided with pieces of potentially valuable knowledge about concept hierarchies among items in the database. Second, this knowledge could be used again to mine for multiple-level or generalized association rules, especially those hierarchical relationships that the expert verifies or wants to test for their validity.

The knowledge that two items are mutually exclusive can be further analyzed in order to decide whether the two products could be included in the same level of taxonomy and under the same parent

node. Searching for mutually exclusive pairs of items in a “blind” manner would produce a huge amount of candidate pairs. Moreover, most of the discovered mutually exclusive items would be uninteresting. The intuition behind searching for mutually exclusive items between the extensions of frequent itemsets is twofold. First, the search space is reduced sensibly, and second, a frequent itemset represents a class of consumers that have particular preferences. So, the mutually exclusive items found in the subspace of a frequent itemset are probably more interesting. Let, for example, consider a large store that sells cloths, shoes, sportswear, etc. Let also, consider itemset A = {Socks, AthleticShoes, Rackets}, that represents athletes who play tennis and itemset B = {Socks, AthleticShoes, Handball}, that represents athletes who play handball. Itemsets A and B may not have high support, while itemset $C = A \cap B$ = {Socks, AthleticShoes} is more likely to have higher support, since it represents the whole class of athletes. However, the two items, Rackets and Handball are expected to be locally frequent in the subset of transactions containing C. The use of traditional association rules does not provide the alternative to explore the C-subspace and consequently a large amount of potentially valuable knowledge will remain hidden. Conversely, with our algorithm Rackets and Handball would be discovered as mutually exclusive items indicating a taxonomical relationship.

The taxonomy generation is needed in many domains. Although taxonomies already exist, they may require refinement either because the consumers' preferences change by time, or because new products are produced, or even because some considerations made when the taxonomy was building were false. Moreover, preferences of consumers vary in different locations and consequently the taxonomies are different. For example, a taxonomy used in a supermarket in a highland place, is not appropriate for a supermarket located near the sea. Finally many different taxonomies can be built for a set of items depending on the various viewpoints. Returning to the example, maybe there is not any taxonomy containing Rackets and Handball at the same level and under the same parent node. Probably, a new taxonomy should be created or an old one should be refined.

Now, let’s return to the example of section 4.4. The extracted mutually exclusive pairs of items listed on table 4 are all different types of coffee (espresso, cappuccino and decaffeinated). If we consider the as items of the same level of taxonomy (Figure 5) and replace these items with a single item (the parent node, i.e. coffee), we are able to increase the minimum support threshold in order to acquire stronger association rules (Table 5).

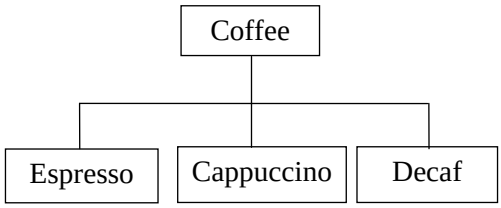

Figure 5: A taxonomy of coffee products

Table 5: Generalized association rules, using the taxonomy discovered

Association Rules	**Support**	**Confidence**
sugar ⇒ coffee	7/9	1
coffee ⇒ sugar	7/9	7/9

CONCLUSIONS

In this paper we have presented the novel problem of mining for mutually exclusive items, as an extension to the association rules mining paradigm. When two items are mutually exclusive, this can be used as a valuable hint when looking for previously unknown taxonomical relationships among them. In such case, this can be an interesting type of knowledge to the domain expert but more importantly, it can be used to mine the database for hierarchical or generalized association rules. For this purpose, we propose an intuitive approach, formulated the problem providing definitions of terms and evaluation metrics. We have also developed a mining algorithm, which extends Apriori, and we have applied it on a large synthetic dataset in order to test its performance. In the future, we would like to apply our approach on real world data sets in order to further investigate its performance.

REFERENCES

Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining Association Rules Between Sets of Items in Large Databases. In Proceedings of the ACM SIGMOD Conference on Management of Data, 207-216.

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. In Proceedings of the International Conference on Very Large Databases. 487-499.

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H. & Verkamo, A.I. (1996). Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Editors), Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park, California, USA. 307-328.

Berberidis, C., Angelis, L., & Vlahavas, I. (2004). PREVENT: An Algorithm for Mining Intertransactional Patterns for the Prediction of Rare Events, In Proceedings of the 2nd European Starting AI Researcher Symposium. 128-136.

Berberidis, C., Tzanis, G., & Vlahavas, I. (2005). Mining for Contiguous Frequent Itemsets in Transaction Databases, In Proceedings of the IEEE 3rd International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications.

Chen, X., & Petrounias, I. (2000). Discovering Temporal Association Rules: Algorithms, Language and System. In Proceedings of the 16th International Conference on Data Engineering.

Han, J., & Fu, Y. (1994). Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases. In AAAI Workshop on Knowledge Discovery in Databases.157-168.

Han, J., & Fu, Y. (1995). Discovery of Multiple-Level Association Rules from Large Databases. In Proceedings of the 21st International Conference on Very Large Databases. 420-431.

Han, J., Pei, J., & Yin, Y. (2000). Mining Frequent Patterns without Candidate Generation. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. 1-12.

Koperski, K. & Han, J. (1995). Discovery of Spatial Association Rules in Geographic Information Databases. In Proceedings of the 4th International Symposium on Large Spatial Databases. 47-66.

Mannila, H., Toivonen, H., & Verkamo, A.I. (1994). Efficient Algorithms for Discovering Association Rules. In Proceedings of AAAI Workshop on Knowledge Discovery in Databases. 181-192.

Savasere, A., Omiecinski, E, & Navathe, S.B. (1998). Mining for Strong Negative Associations in a Large Database of Customer Transactions. In Proceedings of the 14th International Conference on Data Engineering. 494-502.

Srikant, R., & Agrawal, R. (1995). Mining Generalized Association Rules. In Proceedings of the 21st VLDB Conference. 407-419

Teng, C.M. (2002). Learning form Dissociations. In Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery. 11-20.

Thomas, S., & Sarawagi, S. (1998). Mining Generalized Association Rules and Sequential Patterns Using SQL Queries. In Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining. 344-348.

Tung, A. K. H., Lu, H., Han, J., & Feng, L. (2003). Efficient Mining of Intertransaction Association Rules. IEEE Transactions On Knowledge And Data Engineering. 15(1), 43-56.

Wu, X., Zhang, C., & Zhang, S. (2004). Efficient Mining of both Positive and Negative Association Rules. ACM Transactions on Information Systems. 22(3), 381-405.