Frequent Pattern Discovery Without Binarization: Mining Attribute Profiles

Attila Gyenesei[1], Ralph Schlapbach[2], Etzard Stolte[1], and Ulrich Wagner[2]

[1] Knowledge and Data Analysis, Unilever Food and Health Research Institute, 3130 AC Vlaardingen, The Netherlands
{attila.gyenesei, etzard.stolte}@unilever.com
[2] Functional Genomics Center Zürich, Uni ETH Zürich, CH-8057 Zürich, Switzerland
{ralph.schlapbach, ulrich.wagner}@fgcz.ethz.ch

Abstract. Frequent pattern discovery has become a popular solution to many scientific and industrial problems in a range of different datasets. Traditional algorithms, developed for binary (or Boolean) attributes, can be applied to such data with a prerequisite of transforming non-binary (continuous or categorical) attribute domains into binary ones. As a consequence of this binarization, the discovered patterns no longer reflect the associations between attributes but the relations between their binned independent values, and thus, interactions between the original attributes may be lost. In this paper we propose to overcome this limitation by introducing the concept of mining frequent *attribute profiles* that describes the relationships between the original attributes. By this concept, previously hidden interactions can be discovered and redundant patterns that are identified by traditional methods are eliminated. A novel algorithm, called MAP, has been developed for mining attribute profiles that can be potentially applied to diverse data domains. The effectiveness of the proposed method is shown by using gene expression or microarray data.

1 Introduction

The problem of association pattern discovery (APD) originates from market basket analysis which aims at finding interesting relationships hidden in large datasets. Such relationships can be represented in the form of *frequent itemsets* and *association rules*. APD is a two-step process: the first and most time-consuming step is to find frequent sets of items (called itemsets) that occur together in at least as many transactions as a given support threshold. The *support* of an itemset is the number of transactions that contain the itemset. The number of potential frequent itemsets is exponential to the number of items, which presents the main problem of the first step. The second step generates association rules from the frequent itemsets. Based on the mining strategy by which frequent itemsets are discovered, two types of algorithm can be distinguished: breadth-first search and depth-first search. The most commonly used breadth-first search algorithm is Apriori [2,14] and its variants [4,8]; whereas Eclat [16] and FP-growth [9] are popular depth-first search algorithms. For other APD algorithms together with precise descriptions and analyses, see [10].

J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.): PKDD 2006, LNAI 4213, pp. 528 – 535, 2006.

Since its introduction, APD has been successfully applied not only to market basket analysis but to many other scientific and industrial problems, and more recently to gene expression data [5,7,12]. In market basket analysis an item is purchased or not purchased in a transaction, which requires the data to be represented by binary attributes. Real-world datasets, however, often contain continuous and categorical values. In gene expression data, for instance, a real value is assigned to each gene that specifies its expression level in a given tissue or condition. Applying conventional pattern mining algorithms to such datasets requires a preliminary transformation of non-binary attributes to binary ones [15], which can only partly discover association patterns. This limitation of traditional methods can be highlighted by a simple example taken from a gene expression experiment as shown in Table 1. The expression values of gene *A*, gene *B*, gene *C* and gene *D* were determined for four different conditions as significantly repressed (1), significantly expressed (2), or neither significantly repressed nor expressed (3).

Table 1. Sample gene expression data

	gene *A*	gene *B*	gene *C*	gene *D*
cond1	2	3	2	3
cond2	1	2	1	2
cond3	1	1	1	2
cond4	2	1	2	3

The research problem in such data is to find relationships between co-regulated genes, or in other words, to discover frequent combinations and associations of genes that display co-occuring changes condition by condition. Using the traditional methods, first the sample data is transformed into binary data before any of the frequent pattern mining algorithms can be applied. Table 2 summarizes the frequent patterns and their supports for the binned data.

Table 2. Frequent patterns vs. frequent profiles

support	Frequent patterns	Frequent profiles
4	-	$A\{0\}C\{1\}D$ and its 3 subsets
3	-	-
2	{*A*:1, *C*:1, *D*:2}, {*A*:2, *C*:2, *D*:3} and their 6 subsets	$A\{1\}B\{-1\}C\{1\}D$ and its 8 subsets

The highest co-occurrence that can be identified by traditional methods is 50% and it is satisfied by two frequent patterns: {*A*:1, *C*:1, *D*:2} for condition 2 and 3, and {*A*:2, *C*:2, *D*:3} for condition 1 and 4. However, the data in Table 1 show that there is a real association between genes *A*, *C* and *D* for all conditions. Where the expression level of one of the genes is affected in a particular way (repressed, expressed or no change) the expression values of the other two genes are affected in the same way in all conditions. This allows for the identification of genes whose expression profiles

follow the same patterns in response to different conditions. Intriguingly, traditional techniques are unable to identify the relationship even between genes *A* and *C* despite the fact that they have identical expression values in all conditions because of the low support of their binned values. Moreover, with support threshold 2, an accurate method should result in a single pattern for genes *A*, *C* and *D* thus reducing the number of "redundant" patterns (patterns containing the same genes with different binned values).

In this paper we tackle these problems by introducing the concept of mining frequent attribute profiles. Attribute profiles consider the original attributes without the need for binarization, and present their "trends" that are not visible when only binary values are used. By this concept, associations between the original attributes can be discovered that remain hidden to traditional approaches.

Since traditional frequent itemset mining algorithms cannot be applied to discover the introduced attribute profiles, in this paper we propose an efficient depth-first search attribute profile mining algorithm. More precisely, the original data set is first compressed into an attribute distance tree structure where information about the trends (distances) of attributes is stored. Secondly, a recursive searching technique identifies and collects all of the frequent attribute profiles from the attribute distance tree. The effectiveness of the proposed method is demonstrated for a real-world, gene expression dataset in Section 4.

2 Frequent Attribute Profiles

Our starting point is that continuous attributes are only discretized and the data has as many fields as the number of attributes.

Let X and Y be two attributes and let d denote the difference between two attribute values of X and Y in a transaction t such that $d = t[Y] - t[X]$. The formula $X\{d\}Y$ is called an *attribute profile* between attributes X and Y. It is also called *attribute profile of length* 2 since it contains two attributes. The *support* of an attribute profile $p \subseteq P$ in dataset T is the number of transaction that contains the profile in T:

$$supp(p) = |\{\ t \mid p \subseteq t, t \in T\}|$$

The *frequency* of an attribute profile p in T is the probability of p occurring in a transaction $t \in T$:

$$freq(p) = supp(p) / |T|$$

An attribute profile p is called *frequent* if its support (frequency) is greater than or equal to a (user defined) minimum support (frequency) threshold σ_{supp} (σ_{freq}):

$$supp(p) \geq \sigma_{supp} \quad (freq(p) \geq \sigma_{freq})$$

The research problem of attribute profile mining is to find all attribute profiles with sufficient support (frequency). Table 2 summarizes the frequent attribute profiles and their supports for the data given in Table 1.

In contrast to the frequent itemset mining which is unable to identify the association between genes *A*, *C* and *D* in the example from Section 1, attribute profile mining can discover this relation, even when a high (100%) frequency threshold is set. Moreover, we can easily see the trends (distances) of attributes from the frequent attribute

profile. Note that each frequent pattern discovered by traditional methods can be obtained from frequent attribute profiles. For example, the two frequent patterns {*A*:1, *C*:1, *D*:2} and {*A*:2, *C*:2, *D*:3} with support values 2 are included in the attribute profile $A\{0\}C\{1\}D$ with support value 4.

To summarize the advantages of attribute profile mining over frequent itemsets mining, we can conclude that

1. Attribute profiles describe relationships between the entire attributes in contrast to traditional patterns, which identify associations between independent binned attribute values.
2. By the definition of attribute profiles, trends or distances of non-binned attributes are taken into account in order to identify associations between the entire attributes having non-frequent binned values but frequent trends transaction by transaction. Thus, previously hidden (lost) patterns of related attributes can be discovered, which are not found among the patterns produced by traditional APD methods.
3. By discovering attribute profiles, a considerable number of redundant associations can be eliminated. By using traditional methods, redundant associations can appear, when the binned expression values of a set of particular genes have sufficient frequency in more than one condition (or sample).

Note that the distance between two attributes, introduced in this section, can be defined in different ways based on the data domains. Therefore, the distance can be symmetric or asymmetric. Applying symmetric distance measure, patterns of negatively (inversely) correlated genes can be also discovered by attribute profiles that are hidden by traditional patterns. Also note that our distance measure is close to the semantics of functional (multivalued) dependencies, see [1, 13].

3 Mining Attribute Profiles

In this section we present our Mining Attribute Profile algorithm (*MAP*). The proposed algorithm can be characterized as a depth first search, divide-and-conquer algorithm, such as FP-growth [9]. We chose this type of searching strategy in order to reduce the number of database scans and avoid the costly set-containment-test operation that can be the case in applying the breadth-first search strategy, such as Apriori [2].

The mining is carried out in two steps in which the first step constructs a compact data structure called a Frequent Attribute Distance Tree (or FAD-tree), and the second step extracts the frequent attribute profiles directly from this FAD-tree structure.

3.1 Constructing a Frequent Attribute Distance Tree

In order to construct a compact data structure for efficient attribute profile mining, we apply the idea of building a frequent pattern tree (FP-tree) [9]. Similar to the FP-tree, the FAD-tree is constructed by reading the database transaction by transaction and mapping each transaction onto a path in the FAD-tree. A path compression occurs when two or more transactions have the same attribute profiles starting from the first attribute. The main difference between our FAD-tree and the original FP-tree is that

while the FAD-tree is built by the entire attributes (each node is a frequent attribute), the FP-tree is constructed by binned attribute values as single items. Moreover, the FAD-tree considers and stores attribute distances (or trends) between two successive attributes.

3.2 Mining Frequent Attribute Profiles Using the FAD-Tree

The MAP algorithm generates frequent attribute profiles from the constructed FAD-tree by exploring the tree in a top-down and recursive manner. It splits the problem into sub-problems by decomposing the FAD-tree into disjoint sub-FAD-trees and the header table into sub-header tables. This decomposition is carried out attribute by attribute in a stepwise manner. For each attribute node, a parent distance is calculated between the node and the root of the FAD-tree by summing the distance values of intervening nodes. Nodes with calculated equivalent distances are grouped together. For each group with a sufficient support value, a sub-FAD-tree is constructed, i.e. rooted by that attribute. A corresponding sub-header table is also constructed in which the child of the rooted attribute forms the first position. Note that the nodes in the FAD-tree are gathered by using the linked lists in the header table. During the decomposition process, paths with the same parent distance are merged.

The above procedure is applied in a recursive way so that each sub-FAD-tree is used as a FAD-tree in the next recursion. During the decomposition, the roots of sub-FAD-trees are stored as frequent profiles. If the constructed sub-FAD-tree has only a single branch, then there is no need to build a new sub-FAD-tree, all frequent attribute profiles can be enumerated directly from the single branch.

A high-level pseudo-code of the MAP algorithm is given in the following:

Algorithm MAP – Mining Attribute Profiles

Input: FAD-tree *T*, header table *H*, support threshold σ_{supp}
Output: The complete set of frequent attribute profiles
Description:

```
// Check whether T has only a single path.
if T has only a single path then
  enumerate frequent attribute profiles from FAD-tree T
else
  // Main loop for each attribute in the header table
  // with sufficient support value.
  for all attribute a ∈ H, a.supp ≥ σ_supp do
     // Calculate parent distance between the nodes and the
     // root.
     for all n ∈ a.nodes do
        calculate n.parent_dist (the distance between n and
        the T.root)
     Group nodes with same equivalent parent distances
     // For each group with sufficient support, create
     // sub-FAD-tree and sub-header table.
     for each set of a.nodes with same parent_dist and
     sufficient support do
        create sub_FAD-tree sub_T and sub_header table sub_H
        update set of frequent attr. profiles by sub_T.root
        call MAP(sub_T, sub_H, σ_supp)
```

A sub-FAD-tree rooted by attribute *A* is used to demonstrate how the MAP algorithm works for the sample data (Table 1), where the support threshold is set to 2 (Figure 1).

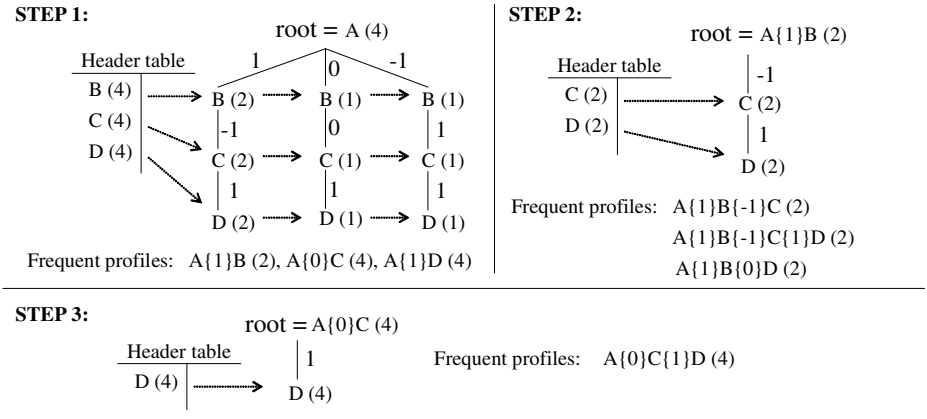

Fig. 1. Applying the MAP algorithm to the FAD-tree rooted by attribute *A*

4 Experimental Results

In this section, we demonstrate the usability and efficiency of our proposed attribute profile mining method (MAP) by comparing the performance of MAP and the traditional frequent itemset mining method (FIM) when applied to the gene expression data set of Hughes et al. [11]. This data contains information about the expression levels of 6316 genes throughout 300 diverse yeast mutants or wild type yeast challenged with different chemical treatments.

We discretized each expression value of the normalized data to 3 integers 1, -1 and 0 representing expressed, repressed and neither expressed nor repressed. This was achieved by assigning all expression values greater than 0.2 for the log base 10 of the fold change to a value of 1 (expressed), all values less than -0.2 to a value of -1 (repressed) and those between -0.2 and 0.2 to 0 (neither expressed nor repressed). To limit the effect of noise, 0 is considered to be a missing value. For FIM, we needed an additional binarization step to bin the gene attributes of discretized data into single Boolean items.

Both FIM and MAP methods were tested on 7 different support thresholds. As an example, here we analyze the lengths of the longest patterns (frequent itemsets and profiles with the greatest number of genes) and the number of maximal frequent patterns. The reason for using the maximal patterns is to consider a hidden association only once, i.e. the longest one (as all subsets of a frequent pattern are frequent as well). Maximal frequent patterns are useful to identify a small representative set of patterns from which all other frequent patterns can be derived. A frequent pattern is called maximal if it has no superset that is frequent [3]. For example, in our sample

dataset, only the listed frequent itemsets and profiles are maximal (Table 2), i.e. having more than one gene. Table 3 summarizes the results for both methods. The frequency threshold values, σ_{rel}, were selected to keep the number of maximal frequent patterns manageable.

Table 3. Comparison of FIM and MAP methods

σ_{rel}	#genes in the longest patterns		# max. patterns (itemsets/profiles)		#max.hidden associations	Running times (in seconds)	
	FIM	MAP	FIM	MAP		FIM	MAP
0.19	4	5	15	163	148	0.16	0.05
0.18	4	6	27	303	276	0.2	0.11
0.17	5	6	48	585	537	0.21	0.12
0.16	6	8	96	1162	1066	0.25	0.19
0.15	7	10	195	2330	2135	0.37	0.35
0.14	8	13	425	4462	4037	1.55	3.45
0.13	11	15	871	7374	6503	3.25	12.56

For all support thresholds, MAP gives the longest and most associations. For example, at a minimum frequency of 0.14 (equal to 42 treatments), the number of genes in the longest associations identified by FIM was 8 whilst MAP identified frequent associations between 13 genes. As a consequence, MAP discovered more maximal frequent patterns than FIM for the same threshold, where all of the associations recognized by FIM were also identified by MAP. The last column shows the number of maximal hidden associations, i.e. the associations that were identified by MAP and not by FIM. The result clearly shows that previously hidden associations can be discovered by our introduced frequent profile mining method.

To compare the running times of traditional FIM and the implemented MAP methods, we chose the FP-growth implementation, provided by Bart Goethals [6]. In situations when higher thresholds are set, MAP is the fastest, whereas in other cases it has the longest running time. However, this is due to the fact that it discovers much more co-regulations between genes for the same thresholds than are found by FIM methods. This is probably a general observation: more frequent profiles and thus, more candidate associations result in slower speed. Similar numbers of profiles and itemsets generate comparable running times. We also wish to point out that our implementation is an initial version with no additional enhancements to increase calculation speed.

5 Conclusions

In this paper, a novel attribute profile mining method was introduced for frequent pattern discovery, as an improvement to the itemset mining approaches common today. The main idea of the method is that non-binary attributes are mined without a preliminarily binarization. As a consequence, frequent patterns of entire attributes hidden to traditional methods can be discovered. An algorithm was developed for the proposed problem and shown to perform effectively when applied to gene expression

data. We expect that the method could also be effectively applied to other large scale data in the area of systems biology, such as protein quantification data, single nucleotide polymorphism data, and data from promoter studies.

References

[1] Agier, M., Petit, J-M., Suzuki, E.: Towards Ad-Hoc Rule Semantics for Gene Expression Data. ISMIS (2005) 494 – 503

[2] Agrawal, R., Srikant, R.: Fast Algorithm for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Data Bases (1994) 487 – 499

[3] Bayardo, R.J.: Efficiently Mining Long Patterns from Databases. Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, (1998) 85 – 93

[4] Brin, S., Motwani, R., Ullman, J., Tsur, S.: Dynamic Itemset Counting and Implication Rules for Market Basket Data. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, (1997) 255 – 264

[5] Creighton, C., Hanash, S.: Mining Gene Expression Databases for Association Rules. Bioinformatics, Vol. 19 (1), (2003) 79 – 86

[6] Frequent Itemset Mining Implementations Repository website, http://fimi.cs.helsinki.fi

[7] Georgii, E., Richter, L., Ruckert, U., Kramer, S.: Analyzing Microarray Data Using Quantitative Association Rules. Bioinformatics, Vol. 21 (2), (2005) 123 – 129

[8] Gyenesei, A., Teuhola, J.: Probabilistic Iterative Expansion of Candidates in Mining Frequent Itemsets. Proc. of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, (2003) 413 – 422

[9] Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without Candidate Generation. Proceedings of the 2000 ACM SIGMOD Int. Conf. on Management of Data, (2000) 1 – 12

[10] Hipp, J., Güntzer, U., Nakhaeizadeh, G.: Algorithms for Association Rule Mining – A General Survey and Comparison. SIGKDD Explorations, Vol. 2(1), (2000) 58 – 64

[11] Hughes, T.R., Marton, M.J., Jones, A.R., Roberts, C.J., et al.: Functional Discovery via a Compendium of Expression Profiles. Cell, Vol. 102, (2000) 109 – 126

[12] Ji, L., Tan, K.L.: Mining Gene Expression Data for Positive and Negative Co-Regulated Gene Clusters. Bioinformatics. Vol. 20 (16), (2004) 2711 – 2718

[13] Mannila, H., Toivonen, H.: Levelwise Search and Borders of Theories in Knowledge Discovery. DMKD 1, (1997) 241 – 258

[14] Mannila, H., Toivonen, H., Verkamo, A.I.: Efficient Algorithms for Discovering Association Rules. Proceedings of the 1994 AAAI Workshop on Knowledge Discovery in Databases, (1994) 181 – 192

[15] Srikant, R., Agrawal, R.: Mining Quantitative Association Rules in Large Relational Tables. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, (1996) 1 – 12

[16] Zaki, M.J.: Scalable Algorithms for Association Mining. IEEE Transactions on Knowledge and Data Engineering, Vol. 12 (3), (2000) 372 – 390