Proceedings of the 9th International Conference on Applied Informatics
Eger, Hungary, January 29–February 1, 2014. Vol. 1. pp. 85–92
doi: 10.14794/ICAI.9.2014.1.85

Finding minimal rare itemsets with an extended version of the Apriori algorithm*

László Szathmáry

University of Debrecen, Faculty of Informatics, Department of IT
Szathmary.L@gmail.com

Abstract

In this paper we investigate the extraction of rare itemsets, focusing on a special subset of rare itemsets called *minimal rare itemsets*. Rare itemsets are important patterns that have a wide range of practical applications, especially in the analysis of biomedical data. In this preliminary study we investigate how to modify the well-known frequent itemset mining algorithm *Apriori* in order to extract rare itemsets too.

1. Introduction

Pattern mining techniques are designed for extracting interesting and useful itemsets from a database. Among them, frequent itemsets are usually thought to unfold "regularities" in the data, i.e. they are the witnesses of recurrent phenomena and they are consistent with the expectations of the domain experts. In some situations however, it may be interesting to search for "rare" itemsets, i.e. itemsets that do not occur frequently in the data (contrasting frequent itemsets). These correspond to unexpected phenomena, possibly contradicting beliefs in the domain [1, 2]. In this way, rare itemsets are related to "exceptions" and thus may convey information of high interest for experts in domains such as biology or medicine.

In this paper we investigate how to use the classical frequent itemset mining algorithm *Apriori* for extracting a special subset of rare itemsets namely the so-called minimal rare itemsets (mRIs).

2. Background

Consider the following sample dataset: $\mathcal{D} = \{(1,\ ABDE),\ (2,\ AC),\ (3,\ ABCE),\ (4,\ BCE),\ (5,\ ABCE)\}$ of size 5×5. Throughout the paper, we will refer to this

*The publication was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

example as "dataset $\mathcal{D}$".

We consider a set of *objects* or *transactions* $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$, a set of *attributes* or *items* $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$, and a relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$. A set of items is called an *itemset*. Each transaction has a unique identifier (*tid*), and a set of transactions is called a *tidset*. The tidset of all transactions sharing a given itemset X is its *image*, denoted by $t(X)$. For instance, the image of $\{A, B\}$ in $\mathcal{D}$ is $\{1, 3, 5\}$, i.e., $t(AB) = 135$ in our separator-free set notation. The *length* of an itemset X is $|X|$, whereas an itemset of length i is called an i-itemset. The (absolute) *support* of an itemset X, denoted by $supp(X)$, is the size of its image, i.e. $supp(X) = |t(X)|$.

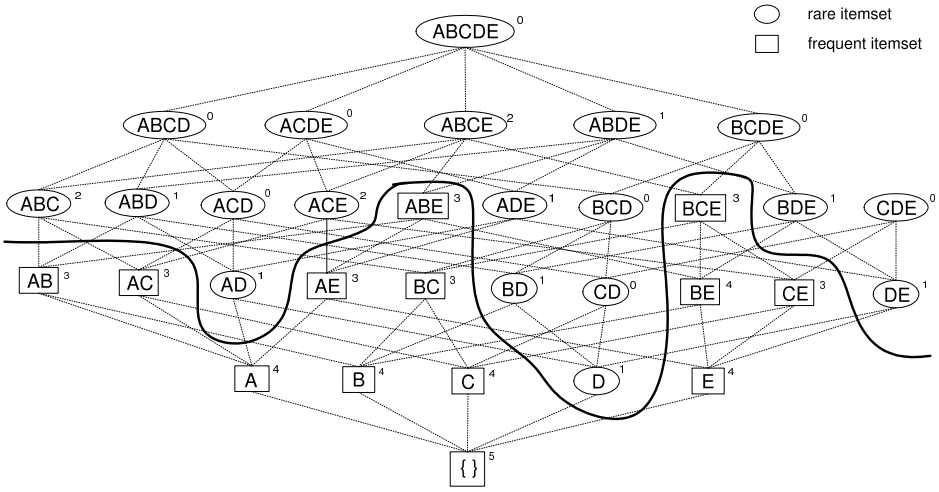

Figure 1: Powerset lattice of dataset $\mathcal{D}$. Support values are indicated in the top right corners. The value of *min_supp* is 3.

Support is a prime measure of interest for itemsets: one is typically – but not exclusively – interested in regularities in the data that manifest in recurring patterns. Thus, intuitively, the itemsets of higher support are more attractive. Formally, the frequent itemset mining assumes a search space for interesting patterns that correspond to the Boolean lattice $\mathcal{B}(2^{\mathcal{A}})$ of all possible itemsets (see Figure 1). The lattice is separated into two segments or zones through a user-provided "minimum support" threshold, denoted by *min_supp*. Thus, given an itemset X, if $supp(X) \geq min\_supp$, then it is called *frequent*. Dually, if a maximal support threshold *max_supp* is provided, then an itemset P such that $supp(P) \leq max\_supp$ is called *rare* (or *infrequent*).

Frequent itemsets (FIs) and rare itemsets belong to two mutually complementary subsets of the powerset $2^{\mathcal{A}}$ that further represent contiguous zones of the lattice $\mathcal{B}(2^{\mathcal{A}})$. In the technical language of lattice theory [3], these zones represent an *order ideal* (or *downset*) and an *order filter* (or *upset*), respectively, which means

that a subset of a frequent itemset is necessarily frequent and, dually, a superset of a rare itemset is necessarily rare. For example, in dataset $\mathcal{D}$ if the minimum support is 3, the itemsets A, AB, or BE are frequent whereas AD or ACE are rare (see Figure 1).

Each of the frequent and rare zones is delimited by two subsets, the maximal elements and the minimal ones, respectively. For instance, the minimal frequent itemset is the empty set (whose support is $|\mathcal{D}|$) whereas the family of maximal frequent itemsets depends on *min_ supp*. Similarly, the unique maximal rare itemset is $\mathcal{I}$ which usually, but not invariably, has support 0.

The above intuitive ideas are formalized in the notion of a border introduced by Mannila and Toivonen in [4]. According to their definition, the maximal frequent itemsets constitute the *positive border* of the frequent zone[1] whereas the minimal rare itemsets form the *negative border* of the same zone.

We consider here a computation of the rare itemsets that approaches them from the frequent zone. Hence we need a characterization of what is widely known as the positive and the negative border of the frequent itemsets, and corresponds for us to the negative *lower* border and the positive *lower* border of the rare itemsets, respectively. Moreover, should one need more than simply the rare itemsets on the border, the adverse *upper* border must be characterized as well.

First, the negative lower border of rare itemsets is a structure known from the literature. The characterization of its members, the *maximal frequent itemsets*, is straightforward:

Definition 2.1. An itemset is a *maximal frequent itemset* (MFI) if it is frequent but all its proper supersets are rare.

Second, the positive lower border of rare itemsets, i.e. the set of minimal rare itemsets is defined dually:

Definition 2.2. An itemset is a *minimal rare itemset* (mRI) if it is rare but all its proper subsets are frequent.

3. Levelwise Algorithms and Apriori

In this section, we present levelwise algorithms in a general way. The most well-known algorithm of this kind, without doubt, is *Apriori* [5]. This algorithm addresses the problem of finding all frequent itemsets in a dataset. *Apriori* has been followed by lots of variations, and several of these levelwise algorithms concentrate on a special subset of frequent itemsets, like closed itemsets or generators. Mannila and Toivonen provided a general framework for levelwise algorithms in [4].

The levelwise algorithm for finding all FIs is a breadth-first, bottom-up algorithm, which means the following. First it finds all 1-long frequent itemsets[2], then at each i^{th} iteration it identifies all i-long frequent itemsets. The algorithm stops

[1] The frequent zone contains the set of frequent itemsets.

[2] That is, first it identifies all frequent items (attributes).

when it has identified the largest frequent itemset. Frequent itemsets are computed iteratively, in ascending order by their length. At each iteration one database pass is needed to count support values, thus the number of database passes is equal to the length of the largest frequent itemset. This approach is very simple and efficient for sparse, weakly correlated data. The levelwise algorithm is based on two basic properties.

Property 3.1 (downward closure)**.** *All subsets of a frequent itemset are frequent.*[3]

Property 3.2 (anti-monotonocity)**.** *All supersets of a non-frequent itemset are non-frequent.*

Consider our previously mentioned toy dataset $\mathcal{D}$. By defining a *min_supp* value, the itemsets can be grouped in two sets: frequent and non-frequent (or rare) itemsets. Between the two sets a border can be drawn that cuts the itemset lattice in a positive and a negative space. Frequent itemsets are on the positive side of the border, while rare itemsets are on the negative side of the border. (The concept of the border theory was introduced in [4]).

Levelwise Exploration of the Positive Side of the Border

The levelwise algorithm discovers frequent itemsets of the itemset lattice in a levelwise manner, i.e. at each level i it only uses i-long frequent itemsets to generate their $(i+1)$-long supersets. These supersets are called *candidates*, and only potentially frequent itemsets are kept. For storing itemsets, two kinds of tables are used: F_i for i-long frequent itemsets, and C_i for potentially frequent i-long candidates. An itemset of length $(i+1)$ is called *potentially frequent* if all its i-long subsets are frequent. Otherwise, if it has an i-long subset not present in F_i, then it means that it has an infrequent subset, and by Property 3.2 the candidate is also infrequent and can be pruned. With one database pass the support of potentially frequent candidates is counted, and itemsets that turn out to be infrequent are removed. The frequent $(i+1)$-long itemsets are used then to generate $(i+2)$-long candidates, etc. The process continues until no new candidates can be generated. The generation of $(i+1)$-long potentially frequent candidates from i-long frequent itemsets consists of two steps. First, in the *join* step, table F_i is joined with itself. The union $p \cup q$ of itemsets $p, q \in F_i$ is inserted in C_{i+1} if they share their $i-1$ first items. Next, in the *prune* step, candidates in C_{i+1} are deleted if they have an i-long subset not present in F_i. This way infrequent itemsets are pruned (it is not necessary to count their supports), and only potentially frequent candidates are kept in C_{i+1}.

Note that the itemsets that are potentially frequent candidates, but turn out to be infrequent, form the set of *minimal rare itemsets*. A minimal rare itemset is a rare itemset such that all its proper subsets are frequent (and necessarily, all its supersets are not frequent). The set of minimal rare itemsets is also known as

[3]The name of the property comes from the fact that the set of frequent itemsets is closed w.r.t. set inclusion.

negative border [4]. By the two basic properties, the levelwise algorithm guarantees that after having found a minimal rare itemset, it will not generate any of its supersets later. As a consequence, supports of proper supersets of minimal rare itemsets will never be counted, and this is the way that the levelwise algorithm reduces the search space in the powerset lattice.

As an example of the join step, consider the F_2 table of dataset $\mathcal{D}$: {AB, AC, AE, BC, BE, CE}. After the join step, C_3 is: {ABC, ABE, ACE, BCE}. Since all their 2-long subsets are present in F_2 (which means that all their subsets are frequent), they are also potentially frequent and kept in C_3. That is, the prune step will not delete any elements of C_3.

The candidate generation and the support counting process require a subset test. In candidate generation, having an $(i+1)$-long candidate, we need to identify its subsets in F_i. In the support counting process, the dataset is read object by object. We need to find the subsets of the corresponding itemset of each object (i.e. the itemset that is included by the given object) in C_k, and the support value of each subset in C_k must be incremented by 1. As it can be seen, these subset operations must be performed *lots of* times, thus it must be implemented in a very efficient way for a good performance. There are two commonly used data structure for subset test: hash-tree, as suggested in [5], and trie (a.k.a. prefix-tree) [6, 7]. Note that for all of our implementations of levelwise algorithms we have used the trie data structure.

4. The R-Apriori Algorithm

It may be surprising, but the easiest way to find minimal rare itemsets is to use the well-known *Apriori* algorithm [8]. *Apriori* is based on two principles (see Properties 3.1 and 3.2). *Apriori* is designed to find all frequent itemsets, but as a "side effect" it also explores the minimal rare itemsets. When *Apriori* finds a rare itemset, it will not generate later any of its supersets because they are surely not frequent. Since *Apriori* explores the itemset lattice level by level from the bottom up, it will count the support of the minimal rare itemsets. These itemsets are pruned, and later the algorithm notices if a candidate has a minimal rare subset (actually *Apriori* checks if all $(k-1)$-long subsets of a k-candidate are frequent. If one of them is not frequent, then the candidate is surely rare. But it also means that the candidate has a minimal rare subset). Thanks to this pruning technique, *Apriori* can significantly reduce the search space in the itemset lattice.

In order to save minimal rare itemsets, *Apriori* needs just a slight modification. If the support of a candidate is less than the minimum support, then instead of deleting it we will save it in the set of minimal rare itemsets (see *R-Apriori*, Algorithm 1).

`SupportCount` method: counts the support of the candidate itemsets. This requires one database pass.

`Apriori-Gen` function: using frequent k-long itemsets it generates potentially frequent $(k+1)$-long candidates. Potentially frequent means that the candidates

Algorithm 1 (R-Apriori):

Description: modification of Apriori to find minimal rare itemsets (mRIs)
Input: dataset + min_supp
Output: all frequent itemsets + minimal rare itemsets

1) $C_1 \leftarrow \{1\text{-itemsets}\}$
2) $i \leftarrow 1$
3) while $(C_i \neq \emptyset)$ {
4) SupportCount(C_i)
5) $R_i \leftarrow \{r \in C_i \mid \text{support}(r) < min\_supp\}$ // *R – for rare itemsets*
6) $F_i \leftarrow \{f \in C_i \mid \text{support}(f) \geq min\_supp\}$ // *F – for frequent itemsets*
7) $C_{i+1} \leftarrow$ Apriori-Gen(F_i) // *C – for candidates*
8) $i \leftarrow i + 1$
9) }
10) $I_{MR} \leftarrow \bigcup R_i$ // *minimal rare itemsets*
11) $I_F \leftarrow \bigcup F_i$ // *frequent itemsets*

have no rare subset, i.e. they have no minimal rare subset. Including a rare itemset means being rare (by Property 3.2).

The execution of the algorithm on dataset $\mathcal{D}$ with minimum support 3 (60%)[4] is illustrated in Table 1. Taking the union of the R_i tables the algorithm finds the minimal rare itemsets (collected in Table 2).

5. Future work and conclusion

The two algorithms, *Apriori* and *R-Apriori*, were implemented in Java. Preliminary tests show that the algorithms perform equally. This is not surprising since *R-Apriori* is just a slight modification of *Apriori*.

As a next step, we are going to test the algorithms on different datasets. We are also curious to find a more efficient solution than *Apriori*. As we know, *Apriori* needs to find all FIs, which can be a huge set on a real-life dataset. How could we reduce the search space? Can we extract rare itemsets directly, i.e. without extracting first frequent itemsets? In the near future we are going to study these questions.

[4]This is equivalent to maximum support 2 (40%).

C_1	supp
{A}	4
{B}	4
{C}	4
{D}	1
{E}	4

R_1	supp
{D}	1

F_1	supp
{A}	4
{B}	4
{C}	4
{E}	4

C_2	supp
{AB}	3
{AC}	3
{AE}	3
{BC}	3
{BE}	4
{CE}	3

R_2	supp
$\emptyset$	

F_2	supp
{AB}	3
{AC}	3
{AE}	3
{BC}	3
{BE}	4
{CE}	3

C_3	supp
{ABC}	2
{ABE}	3
{ACE}	2
{BCE}	3

R_3	supp
{ABC}	2
{ACE}	2

F_3	supp
{ABE}	3
{BCE}	3

C_4	supp
$\emptyset$	

Table 1: Execution of *R-Apriori* on $\mathcal{D}$ with $min\_supp = 3$ ($max\_supp = 2$).

set	supp
{D}	1
{ABC}	2
{ACE}	2

Table 2: Minimal rare itemsets found in dataset $\mathcal{D}$.

References

[1] Liu, H., Lu, H., Feng, L., Hussain, F.: Efficient Search of Reliable Exceptions. In: Proceedings of the 3rd Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining (PAKDD '99), London, UK, Springer-Verlag (1999) 194–203

[2] Suzuki, E.: Undirected Discovery of Interesting Exception Rules. International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI) **16**(8) (2002) 1065–1086

[3] Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. 2nd edn. Cambridge University Press (2002)

[4] Mannila, H., Toivonen, H.: Levelwise Search and Borders of Theories in Knowledge Discovery. Data Mining and Knowledge Discovery **1**(3) (1997) 241–258

[5] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. In: Advances in knowledge discovery and data mining. American Association for Artificial Intelligence (1996) 307–328

[6] Aho, A., Hopcroft, J.E., Ullman, J.D.: Data structures and algorithms. Addison Wesley (1985)

[7] Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. Inf. Syst. **24**(1) (1999) 25–46

[8] Mannila, H., Toivonen, H.: Multiple uses of frequent sets and condensed representations. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96), Portland, USA, AAAI Press (1996) 189–194