Towards Rare Itemset Mining

Laszlo Szathmary, Amedeo Napoli
LORIA, Campus Scientifique B.P. 239,
54506 Vandœuvre-lès-Nancy Cedex, France
{szathmar, napoli}@loria.fr

Petko Valtchev
Dépt. d'Informatique UQAM, C.P. 8888,
Succ. Centre-Ville, Montréal H3C 3P8, Canada
valtchev@iro.umontreal.ca

Abstract

We describe here a general approach for rare itemset mining. While mining literature has been almost exclusively focused on frequent itemsets, in many practical situations rare ones are of higher interest (e.g., in medical databases, rare combinations of symptoms might provide useful insights for the physicians). Based on an examination of the relevant substructures of the mining space, our approach splits the rare itemset mining task into two steps, i.e., frequent itemset part traversal and rare itemset listing. We propose two algorithms for step one, a naïve and an optimized one, respectively, and another algorithm for step two. We also provide some empirical evidence about the performance gains due to the optimized traversal.

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 [12, 16]. 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.

Rare cases deserve special attention because they represent significant difficulties for data mining algorithms [20]. However, the underlying mining problems have not yet been studied in detail. Indeed, the scarce literature on the subject is almost exclusively composed of work on adapting the general levelwise pattern mining framework around the *Apriori* algorithm [2] to various relaxations of the frequent itemset and frequent association notions [11, 21, 9]. Although these methods will typically retrieve large portions of the search space for itemsets and associations that lay outside its frequent part, this coverage nevertheless remains incomplete since many rare associations will not be discovered, either due to an excessive computational cost or to overly restrictive definitions. Hence, as it was argued in [17], these methods will fail to collect a large number of potentially interesting patterns.

As a remedy, we propose a framework that is specifically dedicated to the extraction of rare itemsets. It is based on an intuitive yet formal definition of rare itemset. Its goal is to provide a theoretical foundation for rare pattern mining, with definitions of reduced representations and complexity results for mining tasks, as well as to develop an algorithmic tool suite (within the CORON project [19]) together with the guidelines for its use.

In this paper we present a first result, i.e., a novel method, for computing all rare itemsets. The first step thereof is the identification of the *minimal rare itemsets*. These itemsets jointly act as a minimal generation seed for the entire rare itemset family. In the second step, the minimal rare itemsets are processed in order to restore all rare itemsets. We propose two algorithms for the first step: **(i)** a naïve one that relies on an *Apriori*-style enumeration, and **(ii)** an optimized method that limits the exploration to frequent generators only. The second task is solved by a straightforward procedure. The present paper extends and completes a previous article [18].

The paper is organized as follows. We first provide motivating examples drawn from biomedical applications. Then, we introduce the basic concepts of frequent/rare pattern mining followed by the presentation of our approach with a detailed description of its two steps, the frequent zone traversal in the itemset lattice, and the rare set listing, respectively. Next, we provide the description of the aforementioned three algorithms whereas the frequent traversal algorithms are confronted within an experimental study of the respective performances. Finally, lessons learned and future work are discussed.

2. Motivation

As already mentioned, the discovery of rare itemsets, and in sequence of rare association rules deriving from rare itemsets, may be particularly useful in biology and medicine. Suppose an expert in biology is interested in identifying the cause of cardiovascular diseases (CVD) for a given database of medical records. A frequent itemset such as "{elevated cholesterol level, CVD}" may validate the hypothesis that these two items are frequently associated, leading to the possible interpretation "people having a high cholesterol level are at high risk for CVD". On the other hand, the fact that "{vegetarian, CVD}" is a rare itemset may justify that the association of these two itemsets is rather exceptional, leading to the possible interpretation "vegetarian people are at a low risk for CVD". Moreover, the itemsets {vegetarian} and {CVD} can be both frequent, while the itemset {vegetarian, CVD} is rare.

The second example is taken from the field of pharmacovigilance, i.e., a field of pharmacology dedicated to the detection, survey and study of adverse drug effects. Given a database of adverse drug effects, rare itemset extraction enables a formal way of associating drugs with adverse effects, i.e. finding cases where a drug had fatal or undesired effects on patients. In this way, a frequent association such as "{drug} $\cup$ {A}", where "{A}" is an itemset describing a kind of desirable effect, means that this association describes an expected and right way of acting for a drug. By contrast, a rare itemset such as "{drug} $\cup$ {B}" may be interpreted as the fact that "{B}" describes an abnormal way of acting for a drug, possibly leading to an undesirable effect. Thus, searching for adverse effects for a drug may be stated as a search for rare itemsets in a database.

A third example shows that the extraction of rare itemsets proves to be useful in the analysis of healthy cohorts. The real-world STANISLAS cohort is composed of a thousand presumably healthy French families [15]. The role of the cohort is to record administrative, medical, genetic, and some other general data, in order to carry on experiments, mainly statistical operations. A main objective is to investigate the impact of genetic and environmental factors on the diversity in cardiovascular risk factors. An interesting operation for the domain expert is to extract from the cohort profiles, i.e. itemsets in the present case, associating genetic data with extreme or borderline values of biological parameters. However, such types of associations are rare in a healthy cohort, whence the potential utility of rare itemsets for the expert.

3. Search space structure

Here we recall key notions from the frequent pattern mining field and introduce the notations used in the paper.

3.1. Basic concepts

We assume standard pattern mining settings, i.e., a set of m *items* $\mathcal{I} = \{i_1, i_2, \cdots, i_m\}$ and a *transaction database* (TDB) $\mathcal{D} = \{t_1, t_2, \cdots, t_n\}$ on top of $\mathcal{I}$. A subset X of $\mathcal{I}$ is referred to as *itemset* whereby if $|X| = k$, then X is a k-itemset. Moreover, a *transaction* t is made of an itemset I_t and a unique identifier tid_t, typically, a natural number. A sample TDB made of five transactions is depicted in Figure 1 (left). The fraction of transactions in $\mathcal{D}$ that contains an itemset X is called the (absolute) *support* of X and is denoted by $supp(X) = |\{t | t \in \mathcal{D}, X \subseteq X_t\}|$.

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 $\mathcal{B}(2^{\mathcal{I}})$ of all possible itemsets (see Figure 1, right). 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*, otherwise it is *infrequent* or *rare*.

Frequent itemsets (FIs) and rare itemsets belong to two mutually complementary subsets of the powerset $2^{\mathcal{I}}$ that further represent contiguous zones of the lattice $\mathcal{B}(2^{\mathcal{I}})$. In the technical language of lattice theory [7], 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. In the lattice in Figure 1, the two zones corresponding to a support threshold of 3 are separated by a solid line. For example, the itemsets {A}, {AB}, or {BE} are frequent whereas {D}, {BD}, or {ACD} are rare.[1]

The rare itemset family and the corresponding lattice zone is the target structure of our study. It may be further split into two parts, the itemsets of support zero, hereafter called *zero itemsets* (X with $supp(X) = 0$), on the one hand, and all other rare itemsets, on the other hand. For instance, {BCD} is a zero itemset whereas {D} is a non-zero rare itemset.

It is noteworthy that the overall split of the lattice into three "stripes" depends for its exact shape on the chosen value for *min_supp*. Furthermore, it can be generalized to n stripes by providing an ordered sequence of $n - 2$ values. Typically, we have assumed above that all itemsets can either be rare or frequent, but this needs not to always be the case. Thus, one can have two separate threshold values, one for each family, thus leaving a possibly void intermediate zone of neither-frequent-nor-rare itemsets.

[1] We use separator-free set notations, e.g. {AB} stands for {A, B}.

tid	itemset
1	A, B, D, E
2	A, C
3	A, B, C, E
4	B, C, E
5	A, B, C, E

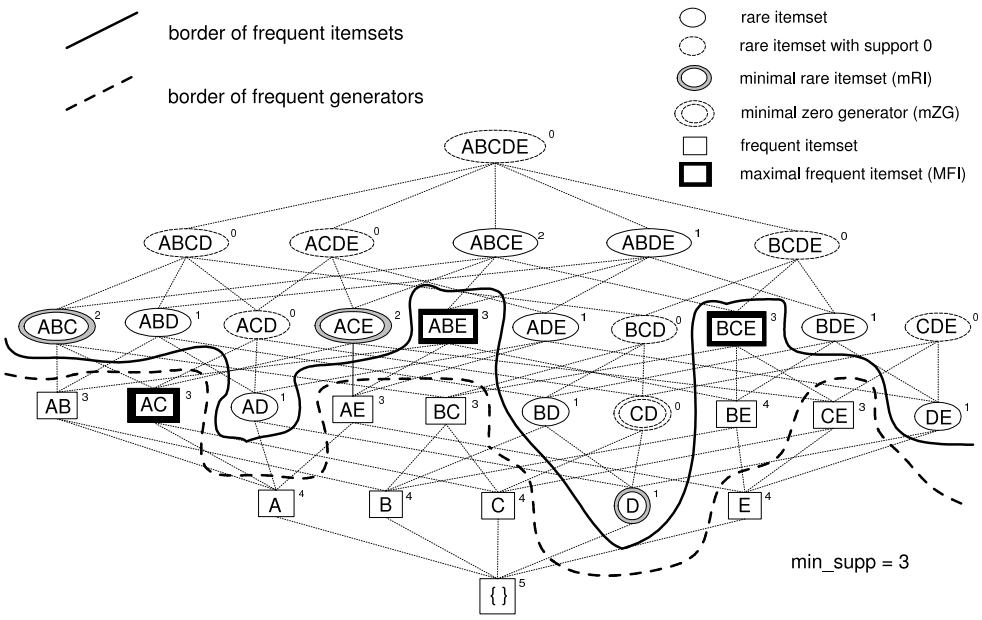

Figure 1. Left: A sample dataset ($\mathcal{D}$) for the examples. Right: The powerset lattice of dataset $\mathcal{D}$.

Whatever the exact number of thresholds and zones, each zone 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 is usually, but not invariably, a zero itemset.

The above intuitive ideas are formalized in the notion of a border introduced by Mannila and Toivonen in [13]. According to their definition, the maximal frequent itemsets constitute the *positive border* of the frequent zone whereas the minimal rare itemsets form the *negative border* of the same zone. Obviously, the same holds for the border between non-zero and zero itemsets as well.

3.2. Computationally motivated results

In order to ground an effective and efficient computation procedure for a particular zone, e.g., the frequent itemset family, one must provide a characterization of its members. Moreover, if the computation is done levelwise, i.e., by visiting iteratively lattice levels that are made of itemsets of a fixed size, one may also need a characterization of the zone border(s). Indeed, if the zone comprises none of the lattice extremal nodes, i.e., $\emptyset$ and $\mathcal{I}$, as is the case of the rare itemset zone, one needs to first pinpoint the starting points of the zone exploration. These starting points are typically the extremal elements, either maximal or minimal, i.e., the positive borders. Furthermore, the computation would typically need to traverse a neighbor zone, hence the negative border of the target zone must also be computed.

We consider here a computation of the rare itemsets that approaches them starting from the lattice bottom, i.e., 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 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 *An itemset is a* minimal rare itemset *(mRI) if it is rare but all its proper subsets are frequent.*

There are at least two possibilities for reaching the mRI family from the lattice bottom node that we discuss in the next section. On the one hand, as we indicated above, a levelwise search listing all frequent itemsets up to the MFIs represents a straightforward solution. Indeed, the levelwise search yields as a by-product all mRIs [13]. On the other hand, the computation of MFIs has been tackled by dedicated methods, hence an alternative solution will be to extract these itemsets directly and then use them as starting

point in the computation of the mRIs, e.g., using the algorithm in [6]. The latter task is known to be computationally hard as it amounts to computing the minimal transversals of a hypergraph [5].

Hence we prefer a different optimization strategy that still yields mRIs while traversing only a subset of the frequent zone of the Boolean lattice. It exploits the minimal generator status of the mRIs.

Definition 3 *An itemset X is a* (minimal or key) generator *if it has no proper subset with the same support ($\forall Y \subset X$, $supp(X) < supp(Y)$).*

In a way similar to FIs, generators form a downset in $2^{\mathcal{I}}$:

Property 1 *All subsets of a generator are generators [10].*

In Figure 1, the downset of frequent generators is delimited by a dashed line. For instance, knowing that $\{\text{BC}\}$ is a frequent generator, $\{\text{B}\}$ and $\{\text{C}\}$ are necessarily frequent generators too. By Property 1, frequent generators (FGs) can be traversed in a levelwise manner while yielding their negative border as a by-product. Now, it is easy to see that all mRIs are part of the negative border of frequent generators. To that end, it is enough to observe that mRIs are in fact generators:

Proposition 1 *All minimal rare itemsets are generators.*

Thus, while there might well be other elements in the negative border that are not generators, e.g., frequent itemsets other than generators, all mRIs will necessarily lay on this border. More specifically, all the rare itemsets on that border will necessarily be minimal for their zone.

It remains now to provide an efficient criterion for recognizing frequent generators. The following property is a reduction of the initial definition to the immediate predecessors of a generator in the lattice (see [3]):

Proposition 2 *An itemset X is a generator iff $supp(X) \neq min_{i \in X}(supp(X \setminus \{i\}))$.*

The property says that in order to decide whether a candidate set X is a generator, one needs to compare its support to the support of its immediate predecessors in the lattice, i.e., the subsets of size $|X| - 1$. Obviously, generators do not admit predecessors of the same support.

The equivalent of the above results can be established for the upper border of the rare non-zero zone of the lattice. Thus, minimal zero generators can be defined as:

Definition 4 *A* minimal zero generator *(mZG) is a zero itemset whose proper subsets are all non-zero itemsets.*

For instance, in Figure 1 there is only one mZG element, $\{\text{CD}\}$. Finally, it is noteworthy that both sides of the border between frequent and rare itemsets play dual role in their respective zones. Indeed, beside being extremal elements, i.e., maximal and minimal, respectively, they constitute reduced representations for these zones as well. For instance, to extract the entire family of frequent itemsets from the MFIs, one only needs to generate all possible subsets thereof. Conversely, if all rare itemsets, i.e., zero and non-zero ones, are necessary, a dual technique will work that amounts to generating all supersets of mRIs [18]. Should zero itemsets be unnecessary, then minimal zero generators would work as stop criterion: only supersets of mRIs that do not include a minimal zero generator will be kept. Provided the support of these sets is required, it can be easily computed along a single pass through the database.

The next two sections present the two methods for mRI computation and the rare itemset listing procedure.

4. Finding minimal rare itemsets

Both algorithms described here produce the mRIs. However, *Apriori-Rare* lists all frequent itemsets before reaching their negative border whereas *MRG-Exp* explores only the frequent generators.

4.1. Finding mRIs with a naïve approach

As pointed out by Mannila and Toivonen in [13], the easiest way to reach the negative border of the frequent itemset zone, i.e., the mRIs, is to use a levelwise algorithm such as *Apriori*. Indeed, albeit a frequent itemset miner, *Apriori* yields the mRIs as a by-product. The mRIs are milestones in the exploration as they indicate that the border of the frequent zone has been crossed.

The overall principle of *Apriori* is rather intuitive: frequent itemsets are generated levelwise, at each iteration i targeting the itemset of length i, i.e., the i^{th} level above the lattice bottom node. The algorithm generates a set of candidates that are further matched against the TDB to evaluate their support in one database pass per iteration. To avoid redundant checks, two techniques are used: **(i)** candidates at level $i+1$ are generated by joining frequent i-itemsets that share $i-1$ of their items, thus increasing the chance of the result being frequent, and **(ii)** candidates are pruned *a priori*, i.e., before support computing, by eliminating those having a rare subset (of size $i-1$). In doing that, there is no need to explicitly represent rare itemsets: rather, all $i-1$ subsets of a candidate are generated dynamically and their presence in the frequent itemset storage structure is tested (absence means the subset, hence the candidate too, is rare).

Algorithm MRG-Exp:

Description: finding minimal rare generators efficiently
Input: dataset plus min_supp
Output: FGs plus mRGs

1) $CG_1 \leftarrow$ {1-itemsets};
2) SupportCount(CG_1); *//requires one database pass*
3) loop over the rows of CG_1 (c) {
4) c.pred_supp $\leftarrow \emptyset$.supp; *//i.e., c.pred_supp* $\leftarrow |O|$*;*
5) if (c.pred_supp = c.supp) c.key $\leftarrow$ false;
6) else c.key $\leftarrow$ true;
7) }
8) $RG_1 \leftarrow \{\ r \in CG_1 \mid$ (r.key=true) $\wedge$ (r.supp $< min\_supp$) $\}$;
9) $FG_1 \leftarrow \{\ f \in CG_1 \mid$ (f.key=true) $\wedge$ (f.supp $\geq min\_supp$) $\}$;
10) for ($i \leftarrow 1$; true; $i \leftarrow i+1$)
11) {
12) $CG_{i+1} \leftarrow$ GenCandidates(FG_i);
13) if ($CG_{i+1} = \emptyset$) break; *//i.e., break out from the "for" loop*
14) SupportCount(CG_{i+1}); *//requires one database pass*
15) loop over the rows of CG_{i+1} (c)
16) {
17) if (c.pred_supp != c.supp) { *//i.e., if c is a generator*
18) if (c.supp $< min\_supp$) $RG_{i+1} \leftarrow RG_{i+1} \cup \{c\}$;
19) else $FG_{i+1} \leftarrow FG_{i+1} \cup \{c\}$;
20) }
21) }
22) }
23) $G_F \leftarrow \bigcup_i FG_i$; *//frequent generators*
24) $G_{MR} \leftarrow \bigcup_i RG_i$; *//minimal rare generators*

Apriori-Rare is a slightly modified version of *Apriori* that stores the mRIs. Thus, whenever an i candidate survives the frequent $i-1$ subset test, but proves to be rare, it is kept as an mRI. For example, following the execution of *Apriori* on dataset $\mathcal{D}$ (Figure 1, left), we get the following result. In C_1 (the set of 1-long candidates), there are 5 itemsets ({A}, {B}, {C}, {D}, and {E}) of which {D} is rare. In C_2 all itemsets are frequent ({AB}, {AC}, {AE}, {BC}, {BE}, and {CE}). In C_3 ({ABC}, {ABE}, {ACE}, and {BCE}) there are two rare itemsets namely {ABC} and {ACE}. Saving the three rare itemsets, one can obtain the following minimal rare itemsets at the end: {D}, {ABC}, and {ACE}.

4.2. Finding mRIs in an efficient way

Following Proposition 1, we may avoid exploring all frequent itemsets: instead, it is sufficient to look after frequent generators only. In this case, mRIs, which are rare generators as well, can be filtered among the negative border of the frequent generators.

For finding minimal rare generators, we focus exclusively on frequent generators and their downset in the lattice (see Algorithm *MRG-Exp*). Thus, frequent i-long generators are joined to create $(i+1)$-long candidates. These undergo a series of tests. On the one hand, the generator status is established following Proposition 2 with the additional

CG_1	pred_supp	key	supp
{A}	5	yes	4
{B}	5	yes	4
{C}	5	yes	4
{D}	5	yes	1
{E}	5	yes	4

RG_1	supp
{D}	1

FG_1	supp
{A}	4
{B}	4
{C}	4
{E}	4

CG_2	pred_supp	key	supp
{AB}	4	yes	3
{AC}	4	yes	3
{AE}	4	yes	3
{BC}	4	yes	3
{BE}	4	—	4
{CE}	4	yes	3

RG_2	supp
$\emptyset$	

FG_2	supp
{AB}	3
{AC}	3
{AE}	3
{BC}	3
{CE}	3

CG_3	pred_supp	key	supp
{ABC}	3	yes	2
{ABE}	3	—	3
{ACE}	3	yes	2

RG_3	supp
{ABC}	2
{ACE}	2

FG_3	supp
$\emptyset$	

CG_4	pred_supp	key	supp
$\emptyset$			

Figure 2. Execution of the MRG-Exp algorithm.

condition that all subsets of the candidate must be frequent generators. Thus, non-generator frequent itemsets and non-minimal rare itemsets are discarded. Next, frequency test against the TDB is used to separate frequent from (minimal) rare generators.

The above reasoning is partly embedded into the `GenCandidates` function which has three-fold effect. First, it produces the $(i+1)$-long candidate generators, using the i-long frequent generators in the FG_i table. Second, all candidates having an i-long subset which is not in FG_i are deleted. In this way, *non-minimal* rare itemsets are pruned, and only potential generators are kept. Third, the function determines the pred_supp values of the candidates, i.e., the minimum of the supports of all i-long subsets.

Later in the process, the pred_supp is compared to the actual support of a candidate. If both values are different then the candidate is a true generator. Moreover, depending on its support, it is either a frequent generator or a minimal rare one, i.e., an mRI.

The execution of *MRG-Exp* on dataset $\mathcal{D}$ (Figure 1, left) with $min\_supp = 3$ is illustrated in Figure 2. The algorithm first performs one database scan to count the supports of 1-long itemsets. The pred_supp column indicates the minimum of the supports of all $(i-1)$-long frequent subsets. Itemsets of length 1 only have one frequent subset, the empty set. By definition, the empty set is included in every object of the database, thus its support is 100%. Comparing the support and pred_supp values, it turns out that all 1-itemsets are generators. Testing the support values, itemset {D} is copied to RG_1, while the other generators are copied to FG_1. In CG_2 there is one itemset that has the same support as one of its subsets, thus {BE} is not a key

Algorithm Arima:

Description: restoring all (non-zero) rare itemsets from mRIs
Input: dataset plus mRIs
Output: all (non-zero) rare itemsets plus mZGs

1) $mZG \leftarrow \emptyset$;
2) $\mathcal{S} \leftarrow$ {all attributes in $\mathcal{D}$};
3) $i \leftarrow$ {length of smallest itemset in mRI};
4) *//add the smallest itemsets in mRI to C_i:*
5) $C_i \leftarrow$ {i-long itemsets in mRI};
6) $mZG \leftarrow mZG \cup \{z \in C_i \mid \text{support}(z) = 0\}$;
7) $R_i \leftarrow \{r \in C_i \mid \text{support}(r) > 0\}$;
8) while ($R_i \neq \emptyset$)
9) {
10) loop over the elements of R_i (r) {
11) *//in $Cand$ no duplicates are allowed:*
12) $Cand \leftarrow$ {all possible supersets of r using $\mathcal{S}$};
13) loop over the elements of $Cand$ (c) {
14) if c has a proper subset in mZG, then
15) delete c from $Cand$;
16) *//i.e., if c is a superset of a min. zero gen.*
17) }
18) *//no duplicates are allowed in C_{i+1}:*
19) $C_{i+1} \leftarrow C_{i+1} \cup Cand$;
20) $Cand \leftarrow \emptyset$; *//re-initializing $Cand$*
21) }
22) SupportCount(C_{i+1}); *//requires one database pass*
23) $C_{i+1} \leftarrow C_{i+1} \cup \{(i+1)$-long itemsets in $mRI\}$;
24) $mZG \leftarrow mZG \cup \{z \in C_{i+1} \mid \text{support}(z) = 0\}$;
25) $R_{i+1} \leftarrow \{r \in C_{i+1} \mid \text{support}(r) > 0\}$;
26) $i \leftarrow i + 1$;
27) }
28) $I_R \leftarrow \bigcup_i R_i$; *//(all non-zero) rare itemsets*

generator. In the fourth iteration no new candidate is found and the algorithm breaks out from the main loop. When the algorithm stops, all minimal rare generators are found ({D}, {ABC}, and {ACE}).

5. Restoring all rare itemsets

To retrieve all rare itemsets from mRIs, we propose a prototype algorithm called "A Rare Itemset Miner Algorithm" (see Algorithm *Arima*). We do not restore zero itemsets because of their high number. Non-zero rare itemsets are restored from mRIs in a levelwise manner. If a candidate has an mZG subset, then it is a clear zero itemset hence it can be pruned. Thus, minimal zero generators help reducing the search space by filtering zero itemsets during candidate generation.

The execution of *Arima* on dataset $\mathcal{D}$ (Figure 1, left) with $min\_supp = 3$ is illustrated in Figure 3. The algorithm first takes the smallest mRI, {D}, which is non-zero, thus it is copied to R_1. Its 2-long supersets are generated and stored in C_2 ({AD}, {BD}, {CD} and {DE}). With one database pass their supports can be counted. Since {CD} is a zero itemset, it is copied to the mZG list. Non-zero itemsets are copied to R_2. For each rare itemset in R_2, all possible supersets are generated. For instance, from {AD} we can generate the following candidates: {ABD}, {ACD}, and {ADE}. If a candidate has an mZG subset, then the candidate is surely a zero itemset and can be pruned ({ACD}). Potentially non-zero candidates are stored in C_3. Duplicates are not allowed in C_i tables. From mRIs the 3-long itemsets are added to C_3 ({ABC} and {ACE}). The algorithm stops when the R_i table is empty. The union of the R_i tables gives all non-zero rare itemsets. At the end, all mZGs are also collected, so that if necessary, zero itemsets could be easily retrieved from that list. The process is identical to rare set generation from mRIs, hence we skip it.

6. Experimental results

In our experiments we compared *Apriori-Rare* and *MRG-Exp*. The algorithms were implemented in Java in the CORON platform [19]. The experiments were carried out on an Intel Pentium IV 2.4 GHz machine running under Debian GNU/Linux operating system with 512 MB of RAM. All times reported are real, wall clock times as obtained from the Unix *time* command between input and output. For the experiments we have used the following datasets: T20I6D100K, C20D10K, and MUSHROOMS. The T20I6D100K[2] is a sparse dataset, constructed according to the properties of market basket data that are typical weakly correlated data. The C20D10K is a census dataset from the PUMS sample file, while the MUSHROOMS[3] describes mushrooms characteristics. The last two are highly correlated datasets.

The execution times of *Apriori-Rare* and *MRG-Exp* are illustrated in Table 1. The table also shows the number of frequent itemsets, the number of frequent generators, the proportion of the number of FGs to the number of FIs, and the number of minimal rare itemsets.

The T20I6D100K synthetic dataset mimics market basket data that are typical sparse, weakly correlated data. In this dataset, the number of FIs is small and nearly all FIs are generators. Thus, *MRG-Exp* works exactly like *Apriori-Rare*, i.e. it has to explore almost the same search space. The reason why *MRG-Exp* is a bit slower is that *MRG-Exp* determines in addition the pred_supp value of each candidate generator.

In datasets C20D10K and MUSHROOMS, the number of FGs is much less than the total number of FIs. Hence, *MRG-Exp* can take advantage of exploring a much less

[2] http://www.almaden.ibm.com/software/quest/Resources/
[3] http://kdd.ics.uci.edu/

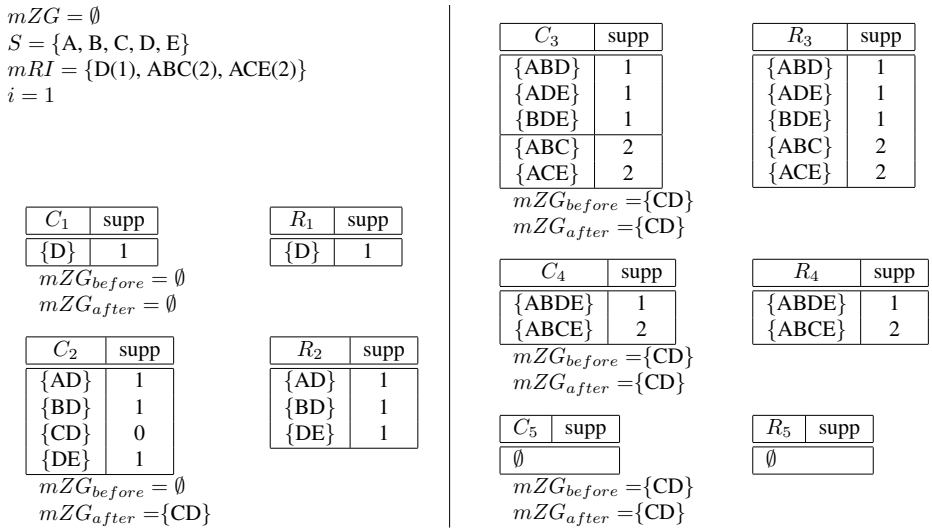

Figure 3. Execution of the Arima algorithm.

search space than *Apriori-Rare*. Thus, *MRG-Exp* performs much better on dense, highly correlated data. For example, on the dataset MUSHROOMS at $min\_supp = 10\%$, *Apriori-Rare* needs to extract 600,817 FIs, while *MRG-Exp* extracts 7,585 FGs only. This means that *MRG-Exp* reduces the search space of *Apriori-Rare* to 1.26%!

7. Related work

Work on various relaxations of the crisp frequent itemset notion has been exclusively based on the *Apriori* algorithm, which is historically the first levelwise pattern mining algorithm [2]. A straightforward extension of the above strategy based on the notion of *perfectly rare* itemset has been used in [9]. Such itemsets have all their non-empty subsets also rare, which means they compose an almost complete downset in $2^{\mathcal{I}}$ (except for the bottom node). Obviously, the perfectly rare itemset family is only a small subset of the entire rare itemset family. Further approaches target the so-called *rare item problem*, and hence try to relax the minimal support threshold for some non-frequent items. As a result, their output will include some of the rare itemsets in the TDB. However, this is by no means a first-class solution to the rare itemset mining problem since: **(i)** only a tiny part of the rare itemset family is retrieved, and **(ii)** both rare and frequent itemsets are mixed.

The tasks of generator and MFI mining have been extensively explored and there is a rich literature on that subject. Historically, *A-Close* [14] was the first algorithm to extract the frequent generators as an intermediate step in the process of frequent closed itemset[4] mining. *A-Close* and *MRG-Exp* share the same principle of limiting the levelwise exploration to frequent generators.

A large number of mining methods target MFIs [4, 1, 8]. Experiments have shown that this approach is very efficient for finding large itemsets in databases.

8. Conclusion and future work

We presented an approach for rare itemset mining from a dataset that splits the problem into two tasks. The first task, i.e., the traversal of the frequent zone in the space, is addressed by two different algorithms, a naïve one, *Apriori-Rare*, which relies on *Apriori* and hence enumerates *all* frequent itemsets; and an optimized one, *MRG-Exp*, which limits the considerations to frequent generators *only*. Experimental results prove the interest of the optimized method on dense, highly correlated datasets. The second task, i.e., the retrieval of all rare itemsets from the minimal ones, is tackled with a straightforward method.

A likely limitation of this study is the need to store all rare itemsets which may prove very expensive in storage space. Furthermore, generating rare association rules from *all* rare itemsets would produce a yet another very large set of association rules. Hence, an intriguing question for future research is the definition of compact representation for rare itemsets and rules, just like the ones defined for frequent itemsets (*closed*, *disjunction-free*, etc.) and rules (*informative* and *generic* bases, etc.).

On the application side, an experimental study with the STANISLAS cohort data is under way and its first results are more than encouraging.

[4] Closed itemsets are the antipodes of generators since maximal for their support value.

min_supp	execution time (sec.)		# FIs	# FGs	$\frac{\#FGs}{\#FIs}$	# mRIs
	Apriori-Rare	MRG-Exp				
T20I6D100K						
10%	11.47	15.91	7	7	100.00%	907
0.75%	146.61	156.65	4,710	4,710	100.00%	211,578
0.5%	238.27	262.32	26,836	26,305	98.02%	268,915
0.25%	586.21	622.30	155,163	149,447	96.32%	537,765
C20D10K						
30%	125.97	26.55	5,319	967	18.18%	230
20%	326.87	50.31	20,239	2,671	13.20%	400
10%	842.85	104.25	89,883	9,331	10.38%	901
5%	1,785.08	162.07	352,611	23,051	6.54%	2,002
2%	4,074.33	228.44	1,741,883	57,659	3.31%	7,735
MUSHROOMS						
40%	13.73	6.00	505	153	30.30%	254
30%	46.10	12.64	2,587	544	21.03%	409
15%	869.27	40.68	99,079	3,084	3.11%	1,846
10%	3,097.16	69.23	600,817	7,585	1.26%	3,077

Table 1. Response times of Apriori-Rare and MRG-Exp.

References

[1] R. C. Agarwal, C. C. Aggarwal, and V. V. V. Prasad. Depth first generation of long patterns. In *Proc. of SIGKDD '00*, pages 108–118. ACM Press, 2000.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In *Advances in knowledge discovery and data mining*, pages 307–328. AAAI, 1996.

[3] Y. Bastide, R. Taouil, N. Pasquier, G. Stumme, and L. Lakhal. Mining frequent patterns with counting inference. *SIGKDD Explor. Newsl.*, 2(2):66–75, 2000.

[4] R. J. Bayardo. Efficiently mining long patterns from databases. In *Proc. of SIGMOD '98*, pages 85–93. ACM Press, 1998.

[5] C. Berge. *Hypergraphs: Combinatorics of Finite Sets*. North Holland, Amsterdam, 1989.

[6] E. Boros, V. Gurvich, L. Khachiyan, and K. Makino. On maximal frequent and minimal infrequent sets in binary matrices. *Annals of Mathematics and Artificial Intelligence*, 39:211–221, 2003.

[7] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2nd edition, 2002.

[8] K. Gouda and M. J. Zaki. Efficiently Mining Maximal Frequent Itemsets. In *Proc. of ICDM '01*, pages 163–170, Washington, DC, USA, 2001. IEEE Computer Society.

[9] Y. Koh and N. Rountree. Finding Sporadic Rules Using Apriori-Inverse. In *Proc. of PAKDD '05, Hanoi, Vietnam*, volume 3518 of *LNCS*, pages 97–106. Springer, May 2005.

[10] M. Kryszkiewicz. Concise Representation of Frequent Patterns Based on Disjunction-Free Generators. In *Proc. of ICDM '01*, pages 305–312, Washington, DC, USA, 2001. IEEE Computer Society.

[11] B. Liu, W. Hsu, and Y. Ma. Mining association rules with multiple minimum supports. In *Proc. of SIGKDD '99*, pages 337–341, New York, NY, USA, 1999. ACM Press.

[12] H. Liu, H. Lu, L. Feng, and F. Hussain. Efficient Search of Reliable Exceptions. In *Proc. of PAKDD '99*, pages 194–203, London, UK, 1999. Springer-Verlag.

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

[14] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. *LNCS*, 1540:398–416, 1999.

[15] G. Siest *et al*. Objectives, Design and Recruitment of a Familial and Longitudinal Cohort for Studying Gene-Environment Interactions in the Field of Cardiovascular Risk: The Stanislas Cohort. *Clinical Chemistry and Laboratory Medicine (CCLM)*, 36(1):35–42, Jan 1998.

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

[17] L. Szathmary. *Symbolic Data Mining Methods with the Coron Platform (Méthodes symboliques de fouille de données avec la plate-forme Coron)*. PhD in Computer Sciences, Univ. Henri Poincaré Nancy 1, France, Nov 2006.

[18] L. Szathmary, S. Maumus, P. Petronin, Y. Toussaint, and A. Napoli. Vers l'extraction de motifs rares. In *Proc. of Extraction et gestion des connaissances (EGC '06), Lille, France*, pages 499–510, 2006. (in French)

[19] L. Szathmary and A. Napoli. CORON: A Framework for Levelwise Itemset Mining Algorithms. In *Suppl. Proc. of the 3rd Intl. Conf. on Formal Concept Analysis (ICFCA '05), Lens, France*, pages 110–113, Feb 2005.

[20] G. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explor. Newsl.*, 6(1):7–19, 2004.

[21] H. Yun, D. Ha, B. Hwang, and K. Ryu. Mining association rules on significant rare data using relative support. *Journal of Systems and Software*, 67(3):181–191, 2003.