Pattern Set Mining Essay: A Long Journey in this Wonderful World

Daniele Di Grandi, student number: 7035616

July 2021

First Part

1 Introduction

Discovering knowledge in data has always been an extremely important process ever since the first Databases appeared. The advantages that this technique entails are not purely about economics and business, but can also embrace other aspects: knowledge is a complicated process, and trying to extrapolate it from existing data that, as human beings, we produce every second, can be a valid method to increase our culture and discover concepts and patterns otherwise inexplicable. Pattern Set Mining (PSM) is exactly the field that describes the task of finding the most frequent and relevant patterns in large Databases, which can be seen as big containers of hidden patterns. However, this task comes with several challenges. First, the computation of all patterns requires a large amount of time. Second, not all patterns are useful or interesting: how can we discriminate and return only the interesting ones? Moreover, what can be an objective definition of interestingness? Several methods were developed to tackle these challenges, from adding constraints on the type of desired patterns, to condense the representation for letting fewer patterns represent more information of the Database. People working in the PSM field developed these solutions in years of hard work and hard thinking, trying to answer to the difficult question "how can we efficiently return the set of patterns that is most appropriate to the current user from a Database?"

Problem definition. Although PSM is utilized in different fields, one of great importance certainly is market basket analysis, which aims at finding regularities in the shopping behaviour of customers of supermarkets and shops in general. In this field, the objective is to find patterns in sets of products - Itemsets - that are frequently bought together. An Itemset is considered to be frequent when it has been bought more than θ times, where θ is defined as a threshold, also called minimal support or *minSup*.

The problem is defined as follows. Let $\mathcal{I}$ be the set of all items in the store, t be a transaction (set of items that have been bought together), D a Database of transactions and I a possible set of items (that is a subset of $\mathcal{I}$ and could be a subset of t). Thus, the objective is to find all the set of items I that are frequent, hence, the number of times that they appear in total over the transactions t has to be greater or equal than the threshold θ. The number of times that an Itemset I appears over all the transactions in the Database is defined as the support of that Itemset I. Hence, if the support of I is greater or equal than the threshold θ (the minimal support), the Itemset I is considered to be frequent.

From a frequent Itemset, several *association rules* can be derived. An association rule is an implication of the form $X \Rightarrow Y$, with disjoint Itemsets X as antecedent and Y as consequent. Moreover, association rules have two main metrics, which are *support*, that is the number of transactions that contain both X and Y over the number of all transactions, and *confidence*, which is the number of times that Y appears in transactions together with X over the total number of transactions containing X.

Where is the problem then? The simplest idea that one can think of is to compute and enumerate all the possible combinations of Itemsets of the items sold in a store. Then, for each Itemset, if it has been bought more than θ times, it is frequent: just simple as that. However, this solution is infeasible: the computation time requires several billion years even for a store that sells only 1000 items. Even by defining a rule to filter out the not interesting Itemsets, the number of returned interesting ones is gigantic. This is known as *pattern explosion*, and the first part of this essay will analyze the most relevant techniques and methods developed to tackle this problem more intelligently.

2 The first steps

In this section all the first steps that were performed to start exploring the PSM area will be explained.

The AIS algorithm. In paper [1], the AIS algorithm is presented in order to solve the problem of mining association rules between sets of items. The AIS algorithm is basically a primitive version of the Apriori algorithm (explained in the next paragraph), in fact, the authors already noticed that "if an Itemset Y is frequent, then every subset of Y will also be frequent".

AIS makes multiple scans of the Database and in each scan, candidate frequent Itemsets are generated. For each transaction, the algorithm determines if that transaction contains the frequent Itemsets discovered in the last scan, then, they are extended with the Itemsets that were expected to be not frequent but turned out to be frequent in the current scan. The authors showed that this analysis was doable, and this is the most important achievement obtained. In the context in which the researchers were working, the algorithm was fast to obtain results, although a better reasoning on the Apriori property would have resulted in a faster algorithm, as explained in paper [2], written by one of the authors that also wrote this paper.

The Apriori algorithm. Aim of paper [2] is to improve the algorithm presented in paper [1]. The Apriori algorithm presented in this paper differs from the AIS algorithm in terms of which candidate Itemsets are counted in a scan and in the way that those candidates are generated. In fact, the Apriori algorithm – although it still makes multiple scans over the Database – finally exploits the real potential behind the Apriori property: candidate Itemsets to be counted in a scan are generated by using only the Itemsets found frequent in the previous scan, because any subset of a frequent Itemset must be frequent as well, given the Apriori property. The result is then an actual *level-wise* search algorithm. Compared to AIS, this procedure results in a generation of a much smaller number of candidate frequent Itemsets.

The results obtained by the authors showed that the Apriori algorithm always outperforms the AIS algorithm, specifically, the performance gap increase with the problem size by a factor of three for small problems to more than an order of magnitude for large problems. However, nowadays the Apriori algorithm is almost always on the slow side compared to modern algorithms for Pattern Set Mining.

A different application field. Paper [3] showed that in order to apply Pattern Set Mining, one does not necessarily need transactions. In fact, the authors of this paper studied through the Pattern Set Mining perspective the sequences of events within a timeframe. In the analysis of sequences, the authors are interested in finding all frequent episodes from a class of episodes. One of the algorithms they described (called WINEPI) is based on the Apriori property: the discovery of episodes is performed by only considering an episode when all its subepisodes are frequent, which are those with a minimal occurrence in a time window (hence, minimal support). The best achievement obtained by this paper is that it has shown for the first time that by accordingly adapting this technique, it is possible to effectively apply PSM to different fields, other than basket case analysis.

3 Constraints

A problem is evident: the number of returned Itemsets is gigantic. One way to limit the number of discovered Itemsets is by formulating more exactly which Itemsets are interesting, through the use of some constraints. This technique is known as *constraint pushing* and the aim is indeed to discover only the Itemsets that satisfy the constraint, instead of applying a filter when all frequent Itemsets have been discovered.

The FP-growth algorithm. Paper [4] proposes a faster alternative algorithm to all the Apriori-like algorithms, which is called FP-growth. In fact, Apriori-like algorithms may suffer in situations with many frequent Itemsets or low minimum support thresholds since it is costly to handle a large set of candidates and repeatedly scan the Database to find them. FP-growth aims to reduce the amount of work required by using a novel data structure: the *frequent-pattern tree*, exploiting the fact that transactions will often share the prefix, since Itemsets that occur most frequently will appear at the beginning of the prefixes of many transactions. Furthermore, because of this structure, the authors were able to completely remove the generation process of candidate frequent Itemsets and they also decrease the number of required Database scans to only two, thus, preparing a tool to be used for constraint pushing. Then, frequent Itemsets are obtained using a pattern growth method, where it is possible to concatenate frequent 1-Itemsets.

Overall, the results are very good, the FP-growth algorithm resulted to be about an order of magnitude faster than Apriori, especially when the Database is dense and/or frequent Itemsets are long.

Does order matter? Paper [5] wanted to explore the idea that certain constraints involving functions such as *median* or *avg*, that exhibit no nice properties – like monotonicity – in general cases, may do so by being *converted* into ones that possess that behaviour in the presence of certain item ordering, with the objective of directly incorporate more constraints to reduce the search space. However, convertible constraints cannot be efficiently pushed deep into the Apriori-like mining but can be readily pushed into the FP-growth algorithm previously described. The paper proposes two algorithms for mining frequent Itemsets with convertible constraints: $\mathcal{FIC}^{\mathcal{A}}$ for the anti-monotone ones and $\mathcal{FIC}^{\mathcal{M}}$ for the monotone ones. The most important difference between the two is that with $\mathcal{FIC}^{\mathcal{A}}$ it is possible to stop searching and prune after an infrequent Itemset is found, thus, actually reducing a large part of the search space, and with $\mathcal{FIC}^{\mathcal{M}}$ one has to search until the very bottom of the tree. Furthermore, is shown that $\mathcal{FIC}^{\mathcal{M}}$ achieves less than 3% runtime benefits in most cases, which is not that

much. In other words, for convertible anti-monotone constraints, the results of this paper obtained with the $\mathcal{FIC}^{\mathcal{A}}$ algorithm are really impressive, but for the convertible monotone ones are less impressive: paper [6] actually proposes an efficient solution for them.

The ExAnte algorithm. As we understood that there is a trade-off between the anti-monotone and monotone constraints, paper [6] is proposing an algorithm – called ExAnte – to efficiently solve this trade-off, and it does that by dealing with monotone constraints. ExAnte is indeed a pre-processing data reduction algorithm that is able to dramatically reduce both the search space and the input Database. In fact, in a time where everybody was searching for a smart way to prune the Itemset lattice to speed up the search, this paper proposes a way to prune the Database, instead of the lattice, which is an extraordinary idea. ExAnte applies two reductions: the α-reduction prunes the *items* that do not satisfy the frequency constraint and the μ-reduction prunes the *transactions* that do not satisfy the given monotone constraint. These two different kinds of pruning cooperate to reduce the search space and the input Database, strengthening each other step by step.

Pushing constraints: do we want to do it? Paper [7] tries to tackle the problem from a different angle: the authors aim at generating all the Itemsets, even if it is a costly operation, since it only needs to happen once. Then, it would be possible to do query mining on this set very efficiently. Thus, the authors argue that pushing constraints is not the best method, because it's based on an out-of-date understanding of the KDD, hence, it should not be used anymore. However, this solution comes with a serious problem: what if this Database of Itemsets no longer fits in memory? The number of obtained frequent Itemsets is already incredibly large, plus, every frequent Itemset can lead to very many association rules, so the number of obtained rules is gigantic. If we have to store gigantic amounts of association rules somehow in a Database and then query that Database, that would be a slow operation. The problem with this paper is that the authors didn't explain where they stored their intermediate results, and, as a result of which, they didn't discuss how feasible this approach is. However, a lot of credit should be given: the authors tried to think about the big picture by reasoning on the nature of the applied solution, while other papers just took an algorithmic problem and started solving it, making the solution always more efficient.

4 Condensed representation

A condensed representation is another way to limit the number of returned Itemsets, and it is a subset of all frequent Itemsets where all the information are there to reconstruct the complete set of frequent Itemsets without looking at the Database.

The Close algorithm. The authors of paper [8] developed the Close algorithm. It is different from existing algorithms, since it is based on the pruning of the closed Itemset lattice in order to find frequent Itemsets. Based on the proof given in the paper that the set of maximal frequent Itemsets M is identical to the set of maximal frequent closed Itemsets MC, it is possible to reduce the amount of work that has to be done, because by having only this condensed representation, it is possible to find the set of all frequent Itemsets using the same threshold as in the complete Database, since the support of an Itemset I is equal to the support of the smallest closed Itemset containing I. Close is not that different from Apriori, the main difference is explained from line 11 of Algorithm 5, which states that if we already have computed the closure, then it is possible to remove the generator. This extra pruning step increases the efficiency of the algorithm, specifically for dense Databases, where Close outperforms Apriori, since it requires fewer scans. However, the authors made an error in Definition 4, which should have been a theorem: you *cannot define* that something is a lattice, you have to *prove* that $\mathcal{L}_{\mathcal{C}}$ is a complete lattice.

But beyond that, the interesting message of the paper is that you get fewer closed frequent Itemsets than there are frequent Itemsets, thus, this is a valid way to tackle the frequent Itemsets explosion.

Non-derivable Itemsets representation. Goal of paper [9] is to reduce redundancies in the set of all frequent Itemsets by only keeping the non-derivable ones, which are the Itemsets for which the support cannot be derived. In fact, through the use of deduction rules, the authors aim at finding non-redundant lower and upper bounds of the support of Itemsets. Furthermore, the rules are proven to be complete, meaning that the obtained bounds will always be tight. The condensed representation returned in this case, is the set of non-derivable Itemsets. The authors explained how to obtain this condensed representation and also a method to derive all frequent Itemsets from the condensed Database. Moreover, the improvement achieved is significant: the reduction of the number of Itemsets obtained with the NDI 36 algorithm, compared to Apriori 36 (as shown in Fig. 5-e of the paper), is pretty impressive. However, still too many frequent Itemsets are returned.

5 How about interestingness?

The pattern explosion problem can be tackled from a different angle: if it would be possible to specify what

makes a set of Itemsets interesting, all one has to do is search for the most interesting set of Itemsets: as simple as that, right? Let's find it out.

The Joint Entropy. Aim of paper [10] is indeed to improve the effectiveness of pattern discovery algorithms by introducing a new framework for mining binary data considering the quality of patterns. The quality of a pattern is measured by the so-called *joint entropy*. This joint entropy is a very interesting measure, based on the concept that information is in the unexpected: highest is the uncertainty, and highest is the amount of information provided. Remember: the interesting news is not the dog biting the owner, but the owner biting the dog: the unexpected.

As an example, if an item occurs in almost every transactions, its uncertainty would be low, because we can expect it to be frequent, hence, also its joint entropy will be low, and the same holds for the opposite scenario, when an item is almost certain not to be frequent. However, when an item occurs in exactly half of the transactions, that would be interesting, and can provide more information on the Database. Hence, the authors wanted to exploit the joint entropy measure to eliminate redundancies in patterns and obtain only patterns that bring different information about the Database. By providing some intuitions, they managed to construct constraints that will return only the interesting patterns. Indeed, the goal was reached, from the large collection of patterns, only those that are viewed as interesting are selected. Moreover, by using this framework it would be possible to do pattern mining also on labelled data.

More metrics, please. Goal of paper [11] is very similar to the one of paper [10], but now with a more targeted focus using more information theory. Hence, the authors provided more quality metrics derived directly from intuition and theory, which should be used as a filter on the returned set of patterns, to obtain the *pattern team*: the best set of patterns according to the quality metrics. Moreover, the interesting bit of this paper, is that not all the quality metrics should be used in the same application: some of them actually describe the opposite thing, and based on the context, the user may choose which metric is adequate for his/her application.

The authors were able to reach their goal, as also shown in Figure 1 of the paper, where with far fewer patterns you get something that is about as good as the original set. Also, as shown in the tables, the heuristic achieved the same max entropy as the other algorithms, in less time.

The chosen few. Paper [12] aims at reducing the number of patterns returned by a data mining operation to a subset that is small enough – turned out to be of size around 30 – to be inspected by a human user while having little redundancy but still providing more or less the same information as the original full pattern set does. In this paper, the information of a pattern set is determined by the partition it induces on the examples, that contain the same set of patterns belonging to the same block. Hence, information is obtained from the composition of all patterns, in contrast to the notion of sets of closed patterns, as proposed by paper [8]. Thus, the idea is that by having a set of patterns, a pattern partitions the transactions by acting as a classifier that indicates which transactions contain a pattern and which don't. Then, next patterns can split again the subgroup into two more subgroups. However, a pattern can also not split the subgroup: if so, that pattern would not be considered interesting.

The idea is very clever and is to be performed as a post-processing step, but the biggest achievement obtained by the authors is showing that for all the results, you don't need very many patterns to have the complete number of partitions.

Tiling. The authors of paper [13] tried to characterize the Database by saying that with every returned pattern, the number of possible Databases that could be the real one, decreases. Pattern mining is still performed, but if we compare the constraint used in this method with "traditional" constraints in frequent Itemset mining, this method has a different and more stringent one, which is based on the concept of a *tile* – block of ones in a 0/1 Database – and its area, that is produced by the Itemset that generates the tile.

Thus, the idea is to only look at things with a large enough area, which can be considered another valid intuition to redefine interestingness itself. However, the concept of a tile was already known in the literature, but the contribution of the authors is that they are looking for *exact tiles*, thus, a collection that completely covers the Database, while most of the research at the time was focusing on studying tiles approximations. In fact, every tile says something about the Database and if one has a complete tiling, then it would be possible to know the Database completely, as a sort of condensed representation. Furthermore, if it would be possible to do this with very few tiles, then they would actually be pretty informative, as shown in Figure 6 – Mushroom tiling – of the paper: with just a few tiles (around 25), 85% of the ones are covered, and with 100 tiles – which is still low – the complete tiling is obtained.

Stop with the enumeration. Since the returned patterns are still too many, the authors of paper [14] tackled the problem from a statistical point of view, by developing a method to generate "random" patterns in such a way that the probability of getting a pattern

depends on either its frequency, area, discriminativity, or squared frequency. Hence, one of the advantages is that the authors were able to not perform an enumeration step before picking the patterns. To do so, they present 4 algorithms to directly sample from the pattern space, in contrast to previous algorithms based on the Markov Chain Monte Carlo (MCMC) method, which were sensitive to the so-called *burn in period*.

Even though the experiments are well performed, what would have been a more interesting experiment would be to give the classifier only the generated patterns and then see how good that works for the classification.

However, as shown in Figure 3 of the paper, it is very interesting that they have sampled based on frequency but somehow – without looking for it – the patterns they discovered have a relatively high feature score, hence, they are probably good for classification. This is a surprising result since it was not the goal of the authors.

The FastEst algorithm. Defining a frequency threshold is always a difficult task. Of course, the defined threshold determines how many patterns will be discovered, which is the number we are trying to decrease. What if it would be possible to define a method that gives the number of discovered patterns as a function of the threshold? Paper [15] explored exactly this idea, by creating a fast and accurate algorithm – called FastEst – to define the number of obtained patterns given a threshold. Indeed, knowing how many patterns will be discovered for each threshold size, is very useful information and helps to decide how low to set the threshold. This algorithm is based on trees, where each node is a frequent Itemset and their children are all its frequent expansions including the items not in the node. The leaves of the tree are the maximal frequent Itemsets already explained in paper [8]. By exploring the tree until a maximal frequent Itemset is found, a path can be sampled. By repeating this step, the average of the sample paths will constitute the final estimation of the number of patterns for a given threshold. By running this algorithm for different threshold values, and by fitting an isotonic regression on the results, it is possible to define the *pattern frequency spectrum*, which provides estimations for the total number of patterns for all possible thresholds.

Table 1 of the paper clearly shows that the FE algorithm is almost always faster than the Exact algorithm giving good results, besides for the "Adults" and "Kosarak" Databases. However, in these and all the other Databases, FE performs faster than the previous algorithm that performs the same task, MCMC.

Self-sufficient Itemsets. An interesting argument is proposed by paper [16], which states that to understand a Database it is enough to obtain the frequent Itemsets, without searching for the association rules within them. Hence, the aim is to minimize the number of discovered Itemsets, by seeing whether or not they are *self-sufficient*. A self-sufficient Itemset is defined by 3 properties, based on what can make it interesting. First, its support should not be lower than expected from any partitioning of the data, thus, it should be productive. Second, it should be non-redundant. Third, it should be productive also to its exclusive domain, that is, it should define its supersets exclusive domain, and we would want to have that their occurrence is larger than expected. However, the fact that only these properties define an interesting Itemset is not convincing. As an example, we can take two frequently bought items, e.g. Pepsi and Coca-Cola. It would have been very interesting to discover that they would never ever bought together, since it could have many implications in real life – such as an optimized design and allocation in supermarkets shelves – but everything in this paper is concerned with being larger than expected, while smaller than expected does not play a role at all. Hence, the properties defined are not the only ones that contribute to define interestingness.

Finally, it is correct that the author used the Holm procedure to adjust the result from the statistical tests, since the objective is to look for significant results and multiple statistical tests are performed.

The other way around: a high p-value? Paper [17] sees each pattern that is discovered as a constraint on how the distribution from where data have been sampled looks like. The aim is to find the smallest set of results that describe the data by using statistical tests as a tool: given a *null model* we want to identify the smallest set of patterns that, when imposed as constraints to the null model, leads to the data no longer being significant. Hence, the approach is that if we have a number of patterns as constraints, that rather than a low p-value – which says that something surprising it's happening – they have a *high* p-value, then we can understand the data, because with a high p-value there is nothing in the data that can surprise us anymore.

The authors formulated this approach in the most general way, but the most interesting result for us is how it performs on finding frequent Itemsets. Figure 1 of the paper shows indeed that very few items are enough to discover a large part of the structure of the data, and the p-value is still not that high.

Hence, the result achieved is to show that with this approach it is possible to find relatively few Itemsets that are together enough to explain most of the variation in the data. In this case, we are still performing multiple statistical tests as in paper [16], but in contrast with it, the adjustment of the result is not

necessary anymore because here we only want to find a minimal number of Itemsets so that our Database is no longer surprising to us, hence, the Database is not significant anymore, and the adjustment is not necessary if you are looking for something insignificant rather than significant.

The MaxEnt distribution. Aim of paper [18] is to design an appropriate model to find the optimal probability distribution subject to some constraints implied by *prior information*, which defines subjective interestingness in a meaningful and practical way. More specifically, the author represents the prior information by the distribution with the largest entropy subject to these constraints on the null model – defined in paper [17] – called the Maximum Entropy (MaxEnt) distribution. The author argued that the MaxEnt distribution is the best choice since it is the most unbiased one, with the result that no undesirable information is injected into the process. It is very interesting that the problem was formulated as a linear programming problem and solved through the use of the Lagrangian multipliers in a straightforward classic method. Moreover, from Figure 4 of the paper, we can conclude that MaxEnt achieved similar but faster results as the method using swap randomization.

Hence, the author found various ways in which the MaxEnt model can be used to contrast patterns in data with prior information, defining multiple subjective interestingness measures, such as the *compression ratio*, which represents how much information is compressed in a pattern.

How to tackle subjectivity. However, interestingness in its pure concept can still be subjective. Hence, the author of paper [18], also wrote paper [19] in order to develop a framework – that for an optimal usage has to be combined with the MaxEnt model – that allows the user to input his/her beliefs and definitions of what makes a pattern interesting, so that the discovered patterns will perfectly match the definition of interestingness of whoever is using this framework, which is theoretically rigorous but practically applicable as well.

Figure 1 of the paper is very good to convey information of what is going on. Basically, in an ideal scenario, from the space of all possible Databases – where the user has some prior beliefs about these Databases – the most surprising patterns are discovered based on the user prior beliefs and given that a pattern holds when there are lots of Databases that do not count, because constrained by the user beliefs. Then, the beliefs are updated, and the process is repeated by giving the most surprising patterns again, until either the Database is completely identified, or the user is satisfied. Since the user cannot express all of his/her beliefs about the world, a model is used instead, which is the *background distribution*. Hence, an initial background distribution models – although not completely – the prior beliefs of the user, and from there, it works the same as explained in the ideal scenario, with the difference that is the background distribution that is being conditioned.

Again, the underlying theoretically concept of subjective interestingness relies on the same concept of the joint entropy seen in paper [10]: the interestingness lies in the unexpected. The step forward made by this paper is to decrease the interestingness found with the unexpected if the pattern discovered is too complex, which is a very reasonable choice.

The KRIMP algorithm. Aim of paper [20] is to find the heuristically best set of frequent Itemsets that compress the Database the best, that is, obtain the set of Itemsets that provide more interesting information about it with the minimum size. To achieve this result, the *Minimum Description Length* (MDL) principle is used. This principle basically says that the size (length) we want to minimize is the sum of the length of the description of the model that we used to encode the data and the length of those encoded data. In order to apply the MDL principle to Itemsets, the authors introduced a *Code Table* which is a two-column table with the Itemsets and their corresponding code. The amount of obtained compression directly depends on these codes. Hence, they have to be chosen carefully. The problem can then be defined as finding the smallest Coding Set such that the total compressed size of the Database with the corresponding Code Table is minimal.

To solve this problem, the authors of this paper introduced a greedy algorithm – called KRIMP – that starts with the singleton Itemsets for the Code Table, and then adds in it one Itemset at a time based on the *Standard Candidate Order*: first descending in support, second descending on the size of the Itemsets, and third lexicographically. Then, the algorithm computes the compression and if it is smaller, it keeps the Itemset, otherwise, it deletes it permanently.

As shown in Table 4 of the paper, the KRIMP algorithm was actually able to reduce the size, for example, in "BMS-wv1" the gigantic number 1531980297 – which is the size of the candidate set of Itemsets – goes back to only 736 in the returned Code Table. Furthermore, Table 8 of the paper shows that is possible to use the Itemsets found by KRIMP to also do classification, which is performed well.

However, are those 736 Itemsets really representative of the Database? The authors provided an answer also to this question, by showing that the KRIMP classifier actually works, and that can only work if the Code Table obtained for each class is a pretty good description of what the distribution in that class looks like.

Second Part

6 Introduction

During the first part of this essay, paper [18] introduced a very interesting approach to tackle the problem of interestingness, that is, through a Linear Programming (LP) model. The reason behind this choice is that LP models are very good when it comes to finding optimal – or near-optimal – solutions for a specific problem.

Furthermore, when it comes to talking about data, nowadays the themes of privacy and data protection take on an important role. Data privacy concerns apply to all sensitive information that organizations handle, including that of customers, shareholders, and employees. Since data have priceless value, they are considered as a precious resource to have and perhaps even hide. Often, these data play a vital role in business operations, development, and finances, in order to obtain competitive advantages over other companies playing in the same market. As a result, exposed sensitive knowledge can constitute a huge problem in terms of monetary losses. In a world where a supermarket like Albert Heijn has in place absurd policies such as the prohibition for customers to take pictures of items due to price competition – yes, personal experience – to try in every possible way to hide important information, it is immediately visible how important this task has become in recent years.

Moreover, data privacy helps to ensure that sensitive data is only accessible to approved parties. It prevents criminals from being able to maliciously use data and helps ensure that organizations meet regulatory requirements. Hence, both the personal as well as the businesses aspect contributes to making the data privacy theme very critical, thus, studied.

The necessity of preserving the privacy of individuals and the need of protecting corporate knowledge has led the researchers to propose a set of techniques that offer sensitive knowledge hiding, and in the second part of this essay, a review on how the power offered by LP – or even Integer Linear Programming (ILP) – models can be exploited to perform a task that is – to an extent – the contrary of Frequent Itemset Mining: *Frequent Itemset Hiding*.

7 (I)LP and its benefits

LP is a method to achieve the best possible outcome for a given problem formulated as a mathematical model with a linear objective function that has to be optimized (minimized or maximized) by leveraging on some decision variables, and under some linear constraints derived by the nature of the problem itself. ILP is the same, but – some of – the decision variables can only assume integer values. This method has several benefits. First, the solution found is guaranteed to be the *best* result, due to the optimality proofs behind the solving algorithms, such as the Simplex for LP models and Cutting Planes or Branch and Bound for ILP models. Second, if the model is very difficult to solve, there is also the possibility to stop the algorithm and return the best solution found so far, or even speed up the whole process with other techniques, such as Column Generation or Local Search. Third, an (I)LP model can be seen as a LEGO construction: as it happens in a LEGO construction that several little pieces are used in different projects to obtain different constructions, (I)LP models can easily be composed/decomposed to better fit the problem that they are trying to solve.

Hence, I think that this approach combined with existing Frequent Itemset Mining techniques and/or Frequent Itemset Hiding purposes can be a very interesting field to explore.

8 Frequent Itemset Hiding

Since we are talking about Itemsets, a typical example of the need for privacy can also involve supermarkets. As previously discussed, supermarkets regularly collect and store in transactional Databases market basket data of their customers' purchases. These organizations may be willing to share their collected information with other parties, such as advertising or consultancy organizations, for mutual benefit. However, supermarkets that share these data want also to protect their hidden secrets that can be exposed if these data will be mined, to prevent business competitors to gain some insights and advantages. Thus, before releasing the data to others, the data holder wants to protect this *sensitive knowledge*.

8.1 A first interesting approach

Paper [21] tackled this task by proposing a novel and exact method for association rule hiding, based on the notion of distance between the original Database and its *sanitized version*, in which all the sensitive association rules have been hidden. The core concept is basically that we accept a small deviation from the original Database, obtaining the sanitized version, that will disallow the production of sensitive Itemsets at a pre-specified support threshold, which guarantee that no sensitive association rules will be exposed. However, during the hiding process, the non-sensitive knowledge must remain as much intact as possible, in order to not lose important information. Therefore, ILP techniques are very well suited to meet this requirement, since that by quantifying the distance between the two Databases, it is possible to construct a minimization problem that minimizes that distance,

making it possible to hide sensitive knowledge while preserving as much as possible the general one.

Due to the closure property of the Apriori algorithm – explained by paper [2] – one can notice that hiding the minimal set of sensitive Itemsets will certainly lead to hiding the set of all sensitive Itemsets, together with their supersets. Counterintuitively, this operation is necessary and it is not a bad thing: suppose the Itemset ab is sensitive and has to be hidden, and abc is a proper superset. If abc is reported as frequent, then all its subsets must be frequent as well, including ab. However, by removing only ab but allowing abc, an important security gap is left, since it is possible to infer the existence of ab from abc.

The procedure. Sensitive Itemset hiding can be considered as a *border revision* process. Given the lattice in Figure 1, it is possible to obtain an initial border – called "original border" – that separates frequent from infrequent Itemsets. Then, by excluding from the former all sensitive Itemsets and their supersets, the original border is updated obtaining the "revised border". In order to hide frequent Itemsets, the original borderline will have to move up in the lattice.

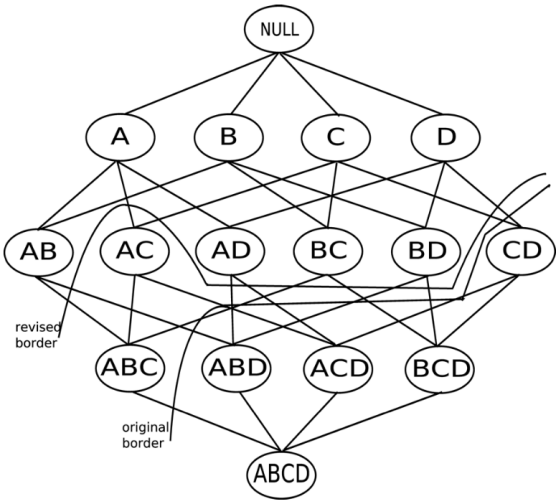

Figure 1: An Itemset lattice with borders.

This procedure is modelled in different steps. The first step is to compute the original border. The authors of this paper decided to utilise the notion of *positive* and *negative* borders. But what are these borders? Suppose that F is the set of all frequent Itemsets in a Database. A negative border is the set of all infrequent Itemsets from that Database in which all proper subsets appear in F, while symmetrically, the positive border of F is the set of all maximally frequent Itemsets appearing in F.

To compute the negative border, the authors decided to use a revised version of the Apriori algorithm, called NB-Apriori. NB-Apriori is based on Apriori's candidate generation scheme which uses $k-1$ frequent Itemsets to produce k candidate frequent Itemsets at level k. In this way, it is possible to identify the first infrequent candidates in the lattice having all their subsets as frequent.

To compute the positive border another algorithm is utilized. This algorithm associate a counter initialized to zero at each frequent Itemset identified using Apriori. Then, the algorithm sorts these Itemsets in decreasing order of length. Suppose the maximum length is k. For all Itemsets of the same length – iterating from k to 1 – the algorithm identifies their $k-1$ frequent subsets and then increases their counter by one. In the end, the algorithm will scan through all the counters searching and collecting the frequent Itemsets having a counter value of zero, and it will add these Itemsets in the positive border.

The negative and positive borders constitute the borderline of the original Database. It is interesting to note that borders form a condensed representation of frequent and infrequent Itemsets in the Database.

As previously mentioned, the next step is to revise this borderline in order to hide sensitive Itemsets and without changing possibly any non-sensitive frequent Itemset to infrequent. The way this error is minimized is indeed by treating this problem as a constraint satisfaction problem and solve it with the help of ILP.

Hence, the third presented algorithm is the one that has to identify the set of all non-sensitive frequent Itemsets by removing the sensitive ones and their supersets from the original set of frequent Itemsets. To do so, the algorithm simply iterates over the set of all sensitive Itemsets and the set of all frequent Itemsets identifying and removing those frequent Itemsets that are supersets of the sensitive ones.

Once the third algorithm has finished its job, by moving top-down in the new lattice it is possible to identify infrequent Itemsets whose all proper subsets are frequent in the sanitized set of frequent Itemset, thus compute the negative border of the new solution.

As a final step of this border revision procedure, the positive border of the new Database is computed using the same methodology previously described. The result obtained if the procedure would be applied on the Database represented by the lattice of Figure 1 is that in the new Database, both Itemsets ac and abc are now hidden.

ILP model. However, what one would like to do in reality is to hide all sensitive Itemsets by minimally affecting the original Database. The problem of affecting the original Database can be summarized in doing two types of errors: Itemsets that appear to be frequent only as a side-effect of the Database sanitization procedure and Itemsets that are no longer frequent always due to that procedure. However, by only

removing items from transactions in the Database, it is ensured that the sanitized version should not have new frequent Itemsets compared to the non-sanitized version. Sadly, there is no such simple reasoning to solve the second type of error. Rather than using the metric of accuracy, as done in the literature before this paper, the authors proposed a new metric measure called *distance*. Although maximizing accuracy is indeed equivalent to minimizing the number of transactions from the original Database that needs to be modified, an algorithm may alter fewer transactions, but in a much higher degree than another one, that probably alters more transactions in a lower degree. Hence, the authors argued that it is the number of actual item modifications that needs to be minimized.

The distance measure between a Database D and its sanitized version D' is defined as follows:

$$dist(D, D') = |\sum_{ij} b_{ij} - \sum_{ij} b'_{ij}| = |\{b_{ij} \neq b'_{ij}\}| \quad (1)$$

Where b_{ij} is equal to 1 *iff* Itemset j is in transaction i in a binary array Database representation, called *bitmap*.

Since in this procedure it is only allowed to remove items from transactions, minimizing the outcome of Equation 1 is the same as maximizing the number of ones left in D':

$$\min\{dist(D, D')\} = \max \sum_{ij} b'_{ij} \quad (2)$$

Equation 2 constitutes the objective function of the ILP problem. However, an ILP problem would not be interesting without constraints. This problem has two constraints. The first one yields that an Itemset X will continue to be frequent in D' *iff* the support of X in D' is greater or equal than the minimum threshold support θ:

$$\sum_{T_i \in D_X} \prod_{I_j \in X} b'_{ij} \geq \theta_{min} \quad (3)$$

The second constraint yields that an Itemset X will be infrequent in the opposite case:

$$\sum_{T_i \in D_X} \prod_{I_j \in X} b'_{ij} < \theta_{min} \quad (4)$$

Hence, all frequent non-sensitive Itemsets must remain frequent in D' due to the constraint expressed by Equation 3. On the contrary, all frequent Itemsets in the set of all sensitive Itemsets should be hidden in D' and therefore they must satisfy the constraint expressed by Equation 4.

As previously said, by using ILP it is possible to obtain suboptimal solutions: this is the case if at least one inequality does not hold for a frequent Itemset.

NP-Hardness. However, the authors of this paper recognized this problem as NP-hard, since in the general case, there are $2^m - 1$ – where m is the number of items – of these inequalities. Fortunately, by noticing that in a typical case there exist many overlapping Itemsets in D and therefore a large number of overlapping inequalities are produced, some of them can be removed. Again, the theme of removing the redundancies – as discussed in the first part of this essay – has come back as a very useful optimization technique. Moreover, another optimization comes from the fact that Equation 3 and 4 do not have the same importance: while it is crucial to ensure that Equation 4 holds for all sensitive Itemsets in D' (in order to be hidden), Equation 3 is there only to enable and improve the border revise process in the Itemset lattice. As a result, the problem was solved through the classic ILP techniques with binary variables and the solution shows very good results in real data, actually being able to hide sensitive knowledge providing an exact solution when it is possible, and a solution with a minimized error when not possible.

Problem. However, this method still has a complexity problem, due to the border idea. Since the Apriori algorithm needs to run first – in exponential time to the number of items of the Database – in order to produce the frequent Itemsets along with their negative border, the complexity is still high. Moreover, the sensitive Itemsets have to be selected from this frequent Itemsets set and then the border has to be revised – which is still a time demanding procedure – before the hiding ILP based procedure can start.

8.2 A substantial improvement

Aim of paper [22] is to solve the just mentioned problem, trying to minimize the preprocessing overhead incurred by border-based techniques. This paper was recently published and it can be considered the state-of-the-art in the Frequent Itemset Hiding field.

The procedure. The novelty introduced by this paper is to model the set of sensitive Itemsets as a *Boolean formula* defined over a set of variables corresponding to the items of the Database. Two are the main advantages. First, the authors can perform a simplification of this formula by using rules from the Boolean Algebra, such as the absorption law to remove redundancies. Second, they can also dispose of the supersets of sensitive Itemsets that need to be neglected upon the computation of the borders. The way this Boolean formula is constructed is simple: every item is mapped to a variable of the Boolean formula and every sensitive Itemset is mapped to a positive term, i.e. a conjunction of positive variables corresponding to the items of the Itemset. The disjunc-

tion of all these terms includes a Boolean formula in Disjunctive Normal Form (DNF) with positive terms.

Also, the authors comprehended that, by removing redundancies, every term of this Boolean formula is minimal and maps a minimal sensitive Itemset.

From this modelling structure, 2 lemmas and a theorem were deducted and proved. The first lemma states that our DNF Boolean formula actually corresponds to the negative border of the sensitive frequent Itemsets set in the border theory. The second lemma states that our negated Boolean formula is equivalent to an irredundant negative DNF Boolean formula. Finally, the theorem states that if we have a non-sensitive frequent Itemset of the ideal sanitized Database, then it corresponds to a negated pattern (a truth assignment with zero values) that satisfies our negated Boolean formula.

Then, the computation of the positive border is performed utilizing the techniques from the constraint-based mining field. As saw in Section 3, the discovery of Itemsets is performed by reducing the number of extracted Itemsets to only those of interest and by pushing constraints inside the mining algorithm to achieve better performances. Hence, these techniques seem to be very well suited for our problem.

Fortunately, all the constraints present in the hiding problem formulation, are anti-monotone ones. In fact, the first constraint in our problem formulation is the support constraint which is satisfied by Itemsets having support greater or equal than the threshold θ, and which is well known to be anti-monotone. The second constraint is related to Itemsets that are non-sensitive, and specifies that an interesting Itemset for our hiding problem can be a proper subset of any sensitive Itemset, but not a sensitive Itemset itself nor a superset of a sensitive Itemset. The authors proved that also this constraint is anti-monotone, by using the fact that this constraint holds for an Itemset only if it satisfies the irredundant negative DNF Boolean formula. Hence, since both constraints are anti-monotone, it is possible to easily push them into the frequent Itemset mining algorithm utilized.

The algorithm. The proposed algorithm that computes the positive border of the Itemsets that satisfy both constraints of the problem works as follow.

The algorithm is initiated by storing all the candidate items that satisfy the irredundant negative DNF Boolean formula in a set C_1. Then, among these items, it stores in a set L_1 the ones found to be frequent after counting them through the first pass in the Database. Then, in the subsequent for-loop, the level-wise operation of the algorithm is performed: the algorithm computes from C_k the candidate Itemsets of the k-th level by applying the Apriori-genB procedure to the set of frequent and non-sensitive Itemsets of size $k-1$, including also the ones placed in L_{k-1}. The Apriori-genB procedure is a modified version of Apriori-gen that performs a 2-way pruning to obtain the new C_k set: the first one is by removing from C_k the candidates of size k with either infrequent or sensitive subsets of size $k-1$, and then it removes sensitive candidates of size k, that are those candidates that do not satisfy the second constraint previously introduced, even if all of their subsets are frequent and non-sensitive. Then, the support of the candidates in the new C_k is computed, and the frequent candidates are placed from C_k into L_k, together with their supports. The algorithm then repeats these steps and terminates when the first empty L_k is obtained, and it returns the union of L_k set collections generated so far, which constitutes the positive border.

Since this algorithm is aimed at improving the efficiency to calculate the borders, it is useful to look at its time complexity, which turned out to be bounded by an exponent to the size of the largest Itemset of the revised negative border. Indeed, the results obtained showed that this constraint-based hiding model always outperforms the already existing level-wise Apriori approach, especially for small supports.

9 Conclusion

In the first part of this essay, the main Pattern Set Mining approaches were presented, while the second part addressed how this field can be extended using Frequent Itemset Hiding techniques, to hide sensitive knowledge that can otherwise be discovered with traditional pattern mining methods.

Although the PSM field still is an interesting area to research, it is also true that it has become highly saturated by several different working solutions, therefore, it also has become highly challenging to come up with innovative ideas. Hence, the interest and effort of researchers can be directed towards PSM correlated topics, trying to solve a myriad of other problems that still need a solution. An example is indeed the Frequent Itemset Hiding problem. Even inside this field, there still are many interesting sub-problems in need of a solution. As in the PSM field researchers came up with innovative techniques to objectively define the subjectivity of what makes a pattern interesting, it would be intriguing to understand if it would be possible to define an objective measure to automatically realize that a frequent Itemset is sensitive, and immediately hide it as soon as it is generated, rather than manually perform this operation.

References

[1] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *SIGMOD Rec.*, vol. 22, p. 207–216, June 1993.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," p. 487–499, 1994.

[3] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Discovery of frequent episodes in event sequences," *Data Min. Knowl. Discov.*, vol. 1, p. 259–289, Jan. 1997.

[4] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *SIGMOD Rec.*, vol. 29, p. 1–12, May 2000.

[5] J. Pei, J. Han, and L. V. S. Lakshmanan, "Pushing convertible constraints in frequent itemset mining," *Data Min. Knowl. Discov.*, vol. 8, p. 227–252, May 2004.

[6] F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi, "Exante: Anticipated data reduction in constrained pattern mining," vol. 2838, pp. 59–70, 09 2003.

[7] J. Hipp and U. Güntzer, "Is pushing constraints deeply into the mining algorithms really what we want? an alternative approach for association rule mining," *SIGKDD Explor. Newsl.*, vol. 4, p. 50–55, June 2002.

[8] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Efficient mining of association rules using closed itemset lattices," *Information Systems*, vol. 24, no. 1, pp. 25–46, 1999.

[9] T. Calders and B. Goethals, "Non-derivable itemset mining," *Data Mining and Knowledge Discovery*, vol. 14, pp. 171–206, 02 2007.

[10] A. J. Knobbe and E. K. Y. Ho, "Pattern teams," in *Knowledge Discovery in Databases: PKDD 2006* (J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, eds.), (Berlin, Heidelberg), pp. 577–584, Springer Berlin Heidelberg, 2006.

[11] A. J. Knobbe and E. K. Y. Ho, "Maximally informative k-itemsets and their efficient discovery," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, (New York, NY, USA), p. 237–244, Association for Computing Machinery, 2006.

[12] B. Bringmann and A. Zimmermann, "The chosen few: On identifying valuable patterns," in *2007 7th IEEE International Conference on Data Mining (ICDM '07)*, (Los Alamitos, CA, USA), pp. 63–72, IEEE Computer Society, oct 2007.

[13] F. Geerts, B. Goethals, and T. Mielikäinen, "Tiling databases," in *Discovery Science* (E. Suzuki and S. Arikawa, eds.), (Berlin, Heidelberg), pp. 278–289, Springer Berlin Heidelberg, 2004.

[14] M. Boley, C. Lucchese, D. Paurat, and T. Gärtner, "Direct local pattern sampling by efficient two-step random procedures," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, (New York, NY, USA), p. 582–590, Association for Computing Machinery, 2011.

[15] M. van Leeuwen and A. Ukkonen, "Fast estimation of the pattern frequency spectrum," in *Machine Learning and Knowledge Discovery in Databases* (T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, eds.), (Berlin, Heidelberg), pp. 114–129, Springer Berlin Heidelberg, 2014.

[16] G. I. Webb, "Self-sufficient itemsets: An approach to screening potentially interesting associations between items," *ACM Trans. Knowl. Discov. Data*, vol. 4, Jan. 2010.

[17] J. Lijffijt, P. Papapetrou, and K. Puolamäki, "A statistical significance testing approach to mining the most informative set of patterns," *Data Mining and Knowledge Discovery*, vol. 28, 12 2012.

[18] T. D. Bie, "Maximum entropy models and subjective interestingness: an application to tiles in binary databases," 2010.

[19] T. De Bie, "Subjective interestingness in exploratory data mining," in *Advances in Intelligent Data Analysis XII* (A. Tucker, F. Höppner, A. Siebes, and S. Swift, eds.), (Berlin, Heidelberg), pp. 19–31, Springer Berlin Heidelberg, 2013.

[20] J. Vreeken, M. Leeuwen, and A. Siebes, "Krimp: Mining itemsets that compress," *Data Min. Knowl. Discov.*, vol. 23, p. 169–214, July 2011.

[21] A. Gkoulalas-Divanis and V. S. Verykios, "An integer programming approach for frequent itemset hiding," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, (New York, NY, USA), p. 748–757, Association for Computing Machinery, 2006.

[22] V. S. Verykios, E. C. Stavropoulos, V. Zorkadis, and A. K. Elmagarmid, "A constraint-based model for the frequent itemset hiding problem," in *E-Democracy – Safeguarding Democracy and Human Rights in the Digital Age* (S. Katsikas and V. Zorkadis, eds.), (Cham), pp. 49–64, Springer International Publishing, 2020.