ExAnte Method for Mining Infrequent Itemsets in Transactional Database

Trapty Jain#1, Megha Kamble#2

1M.Tec scholar(M.tec CSE). Lakshmi Narain College of Technology, Bhopal, India

2Department of Computer Science & Engg,Lakshmi Narain College of Technology, Bhopal, India

Abstract- Many strategies have been introduced to add several types of constraints within the most well known algorithms for mining frequent patterns. The current one algorithm to find frequent items is FP-growth algorithm. Infrequent Itemset mining is a variation of frequent itemset mining where it finds the rare patterns i.e., it finds the data items which occur very rarely. When there is need to minimize a certain cost function, discovering rare data correlations is more interesting than mining frequent ones. The existing method for discovery available in literature but there are some drawbacks related to itemset search space and large input database. The objective of this paper is to overcome these limitations. In this paper FP-Bonsai algorithm is proposed to find infrequent items. FP-Bonsai improve FP-growth performance by reducing (pruning) the FP-tree. In this algorithm, ExAnte data reduction technique is used in which double reduction is applied to find rare patterns. This technique is more efficient than existing methods in the context of reduction of search space i.e. reduction of memory requirement and reduction of large transactional database by applying constraints.

Keywords — Data mining, frequent itemset mining, infrequent mining, FP-tree and constraint mining data mining.

I. INTRODUCTION

Data Mining means extracting information or knowledge from large amount of database. Itemset Mining is an exploratory data mining technique and it is widely used for discovering valuable correlations among data. The first attempt to perform itemset mining was focused on discovering frequent itemsets, i.e., patterns whose frequency of occurrence in the source database (the support) is above a given threshold. There are number of applications of Frequent itemsets in real-life contexts e.g. consumer market basket analysis, biological data analysis, medical image processing, iceberg-cube computation and inference of patterns from web page access logs. Constrained itemset mining is also an active research area in data mining. It finds all transactions/itemsets included in a source database that satisfy a given set of constraints. Frequency constraint is the most studied constraint. Frequency constraint uses a property of anti-monotonicity which reduces the exponential search space of the problem. The frequency constraint uses anti-monotonicity and exploiting anti-monotonicity is known as apriori trick [1, 15] which reduces the search space dramatically as well as making the computation feasible. Since frequency provides “support” to any discovered knowledge so it is not only computationally effective but also it is semantically important. For these reasons frequency is the base constraint in frequent itemset mining.

There are mainly two different types of constraints in constraint itemset mining: anti-monotone constraint and monotone constraint. A constraint CT_{AM} is anti-monotone for any given itemset E, if CT_{AM} holds for E then it also holds for any subset of E and a constraint CT_M is monotone for any itemset E if CT_M holds for E then it holds for any superset of E.

The issue of how to push different types of constraints into the frequent itemsets computation has been extensively studied [16, 17]. However, pushing anti-monotone constraints deep into the mining algorithm is easy and effective but the case is different for monotone constraints.

Indeed, anti-monotone constraints can be used to effectively prune the search space to a small downward closed collection, while the upward closed collection of the search space satisfying the monotone constraints cannot be pruned at the same time.

Recently by using the ExAnte data-reduction technique, it has been shown that a real synergy of these two opposite types of constraints (i. e. monotone constraint and anti-monotone constraint) exists and can be explained by reasoning on both the large input database and itemset search space together. In this way, anti-monotone pruning opportunities do not reduce by pushing monotone constraints. But the opposite, it is boosted up. The two components (that are CT_M and CT_{AM}) strengthen each other recursively like pushing anti-monotone constraints boosts monotone pruning opportunities and vice versa.

In this paper we show how our proposed algorithm can be exploited even better within the well known FP-growth algorithm [18]. The FP-growth computation is done into two phases. During first pass FP-tree is constructed. This construction is done by scan data and finds support of each item in decreasing order and arrange in tree structure. All the frequent patterns trees i.e. FP-tree built recursively and it can be pruned extensively by using the ExAnte property. Finally obtain a smaller number of smaller trees, during computation. We call such a tiny FP-tree, an FP-bonsai which is obtained by growing and pruning. FP-Bonsai improves performance of FP-growth. The resulting method overcomes the main drawback of FP-growth, which is their memory requirement by reducing search space.

FP-tree is a tree like data structure and it is use store frequent items. A FP-tree has root node, prefix sub-tree which has child nodes and frequent item header table. Each node of prefix sub-tree has {item_name, count, node_link}. The item_name indicates to which item this node represent, count indicates number of transactions represented by particular portion and node_link indicates the path of reaching the node. Links to the next node are carrying same item name or null.

In recent years, the main attention of the research community is the infrequent itemset mining problem, i.e., discovering rare itemsets whose frequency of occurrence in the analyzed data is less than or equal to a maximum threshold. Infrequent itemset discovery is applicable to data coming from different real-life application contexts such as- fraud detection where infrequent patterns in financial or tax data may suggest unusual activity associated with fraudulent behaviour, statistical disclosure risk assessment where rare patterns in anonymous census data can lead to statistical disclosure, mining of negative association rules from infrequent itemsets, bioinformatics where rare patterns in microarrays data suggest genetic disorders and in detecting outliers where rare patterns show abnormal behaviour of any event etc. so now-a-days infrequent patterns or rare itemsets has become more interesting area.

II. LITERATURE REVIEW

A. *Techniques used for Frequent Pattern Mining*

1) *Uniform distribution of items:* R.Agarwal[1] introduces Frequent itemset mining which is widely used data mining technique. Here, the rules are generated based on the itemset mined which is said to be frequent. Frequent itemsets are those whose satisfying minimum support and confidence and is used for generating association rules. Most approaches to association rule mining assume that all items within a dataset have a uniform distribution with respect to support. The main problem associated with this is items in a transaction are treated equally.

2) *Significance of item:* In [2]W.Wang introduces the concept of weight to be assigned for item in each transaction which reflects the intensity or the importance of the item within the transaction. The main drawback is that weights are introduced only during the rule generation step not used for the mining purposes.

3) *Weighted Association Rule Mining:* In [3]Feng Tao et.al introduces Weighted Association Rule Mining for frequent itemset mining. In this work the limitation of the conventional Association Rule Mining model is avoided specifically its inability for treating units differently by using weights that describe the local significance of the itemsets and by using new concept of weighted downwards closure property. But the main limitations with these weights are to be pre-assigned which is difficult in real life cases.

4) *Data trimming framework or Apriori Algorithm:* In [4]data trimming framework is presented for mining frequent itemsets from uncertain data under a probabilistic framework. This method uses the U-Apriori algorithm, which is a customized part of the Apriori algorithm, to process on various datasets. Apriori works in two phases. During the first phase it generates all possible Itemsets combinations. These combinations will act as possible candidates. The candidates will be used in subsequent phases. In this algorithm, first the minimum support is applied to find all frequent itemsets in a database and second, these frequent itemsets and the minimum confidence constraint are used to form rules. The main drawback of Apriori is the generation of large number of candidate sets and required more computation time.

5) *FP-Growth* Algorithm:* Grahne et al [6], found that for traversing the FP-trees, 80% of CPU was used i.e. it required more computing time. FP-growth* algorithm uses array based data structure to store FP-tree that incorporates various optimization techniques. Array-based technique is used to reduce the traversal time of FP-tree. The main strength of FP-growth that it reduces the memory consumption as compared to FP-growth Algorithm.

6) *Enhanced FP-Growth Algorithm:* Grahne G.[7] introduced Enhanced FP-Growth algorithm which worked on without any prefix tree or any other complex data structure. It initially scans the supports of the items and is calculated. The items whose support count is less than minimum support are discarded and specified as infrequent items. Then the items in the database are sorted in ascending order with respect to their support. And the initial transaction database is converted in to a set of transaction list, with one list for each item. These lists are stored in array, each of which contains a pointer to the head of the list. And the Transaction lists are traversed from left to right for finding all the frequent item set that contain the item the list corresponds to. Before a transaction list is processed, its support count is checked, if it exceeds than minimum support count than there must be a frequent item set. It processes the

transactions directly, so its main strength is its simplicity.

B. *Techniques use for Infrequent Itemset Mining*

1) *Apriori inverse algorithm:* Yun Sing Koh et al. [8] developed Apriori Inverse algorithm which involves defining both minimum and maximum support threshold for generating a set of infrequent item set. During each iteration, only those items set whose support lies between minimum and maximum support is considered for further processing.

2) *Minimally Infrequent Item set (MINIT):* Haglin et al. [9] presented an algorithm called Minimally Infrequent Item set (MINIT).This was the first algorithm designed for mining minimal infrequent items. The algorithm works by sorting in ascending order of support. Thus minimal infrequent item sets are considered based on the rank and generated using recursive call of MINIT algorithm. This algorithm represents memory efficiency by using pruning.

3) *Minimal Rare Generator (MRG) Algorithm:* Laszlo Szathmary et al. [10] presented an algorithm called Minimal Rare Generator (MRG) to find both rare as well as frequent itemset. Authors used three parameters; they are pre defined support, occurrence of each item and a key. Each itemset is given a predefined support and key with value yes if the items predefined support and support are equal or no otherwise. Only the Itemset with key value yes is considered for next iteration.

4) *Minimally Infrequent Itemset Mining using Pattern-Growth Paradigm and Residual Trees:* Ashish Gupta et al. [11] proposed an algorithm based on pattern growth paradigm to find the infrequent patterns. The algorithm constructs header table for each patterns which is linked to the pattern growth tree containing all item transaction. The authors have used two more structure namely projected and residual trees. Projected tree is constructed by removing the frequent items and residual tree is constructed to reduce space.

5) *Streaming Rare Pattern Tree (SRP):* David Huang et al. [12] proposed a new algorithm called Streaming Rare Pattern tree (SRP) to generate a set of rare items. In this approach, the items in the incoming transaction are inserted into a prefix tree based on FP growth approach. Generally, FP tree is modelled after arranging all items in the transactions in descending order of the support. But in the case of data sets, arranging the items altogether is not possible. To overcome this problem, a structure called connection table is maintained which keeps track of items in the window in canonical order. If an item has support less than minimum support, then the path containing the item generates the all subset of infrequent items.

6) *FP-tree Based Algorithm:* Tsang et al. [13] proposed a FP-tree based algorithm for generating a set of rare items. In this algorithm, the whole transactional database was scanned only once to find rare patterns whose support is less than minimum support.

7) *FP-Growth Algorithms:* Cagliero et al. [5] introduced the idea of Infrequent Weighted Itemset (IWI) and mining Minimal Infrequent Weighted Itemset (MIWI) based on FP- growth approach and both are projection-based algorithms. Hence, it performs the main FP-growth mining steps: (a) FP-tree creation and (b) recursive itemset mining from the FP-tree index. Unlike FP-Growth, IWI Miner discovers infrequent weighted itemsets instead of frequent ones. To accomplish this task, the following main modifications with respect to FP-growth have been introduced: (i) A novel pruning strategy for pruning part of the search space early and (ii) a slightly modified FP-tree structure, which allows storing the IWI-support value associated with each node. The main difference between two algorithms is MIWI Miner focuses on generating only minimal infrequent patterns while IWI miner focuses on both minimal and not minimal patterns, the recursive extraction in the MIWI Mining procedure is stopped as soon as an infrequent item set occurs. It finds both the infrequent item sets and minimal infrequent item set mining.

III. PROPOSED METHOD

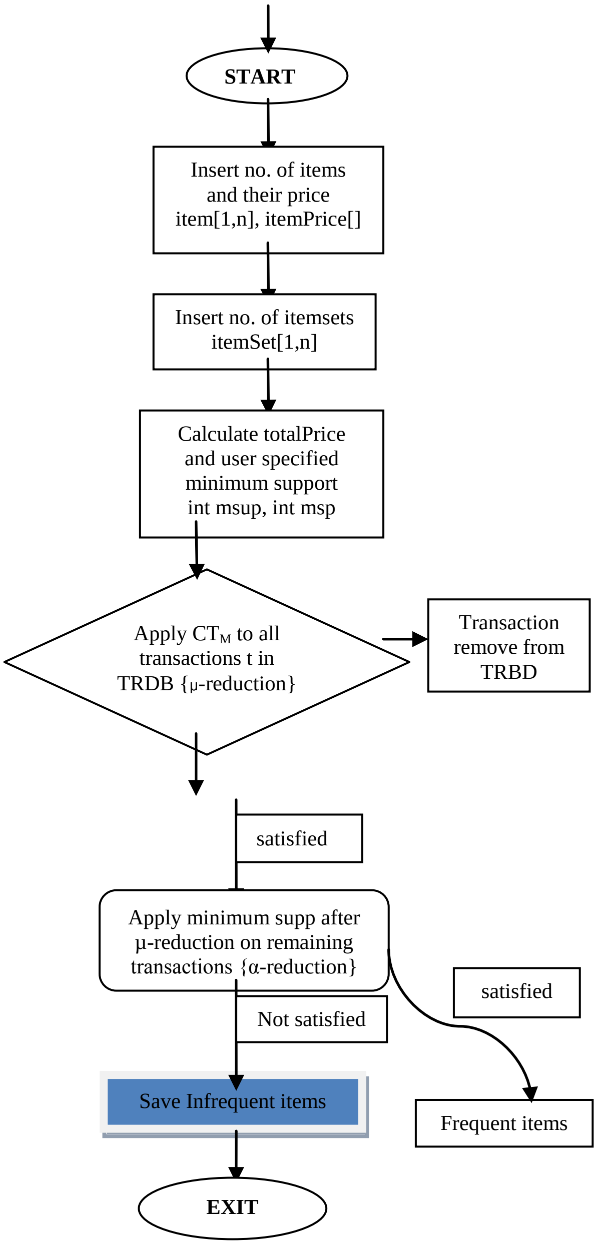
The proposed method introduces ExAnte method [19] to find rare patterns that exploits monotone constraints CT_M in order to reduce large input database and to prune search space. This method is based on the synergy of the following two data-reduction operations: (1) μ-reduction, which deletes transactions in database DB which do not satisfy monotone constraint (CT_M); and (2) α-reduction, which deletes from all transactions in database DB singleton items which do not satisfy support.

The ExAnte property states that a transaction can be removed from the source database which does not satisfy the given monotone constraint CT_M (known as μ-reduction) and it will never consider to the support of any itemset satisfying the constraint.

In this way we find a major result that is reduction of the input database by reduces the support of a large amount of itemsets implicitly. As a result, some singleton items or 1-temset can become infrequent and it can not only be removed from the computation but also they can be deleted from all transactions in the source database or input database (apply α-reduction). This removal of items also has another positive effect. That is, the reduced transaction might violate the monotone constraint CT_M.

The two different type of reduction that are α-reduction and μ-reduction running step by step to prune search space and reduce large input database and continuing until no more reduction is possible. In this way the two reductions are strengthening each other. At last a fix-point has been reached. This is the key idea of the ExAnte pre-processing method.

In the end, the reduced dataset resulting from this fix-point computation is usually much smaller than the initial dataset. Removed items are saved for infrequent mining.


START
Insert no. of items and their price item[1,n], itemPrice[]
Insert no. of itemsets itemSet[1,n]
Calculate totalPrice and user specified minimum support int msup, int msp
Apply CT_M to all transactions t in TRDB {μ-reduction}
Transaction remove from TRBD
satisfied
Apply minimum supp after μ-reduction on remaining transactions {α-reduction}
satisfied
Not satisfied
Save Infrequent items
Frequent items
EXIT

IV. IMPLEMENTATION

- In this section we implement ExAnte algorithm for finding infrequent itemset mining. In the first iteration ExAnte counts the support of 1-itemset i.e. singleton items. This seems similar to any frequent pattern mining algorithm.
- And those items which are not frequent are separated once and for all. But only transactions that satisfy CT_M are selected during this first count. The rest of transactions are signed to be removing from the dataset. This is called *μ*-reduction. In this way we reduce the number of interesting 1-itemsets or singleton itemsets. This small reduction of search space represents a huge pruning.
- At this point ExAnte deletes all infrequent items from alive transactions, this is called *α*-reduction. So we can save these items in an array. The monotone value like total sum of prices in our example of some alive transactions can be reduced by this pruning and possibly resulting in a violation of the monotone constraints. Hence we have *μ*-reduction like other advancement for the dataset. But after *μ*-reduction the dataset we create new chance for α-reduction, which can turn in new opportunities for *μ*-reduction and this process, is continue until we reach on a fix-point.

Understand by example: Suppose that there are some transactions and their price datasets are given [19] in Table 1, Table 2, and Table 3.

- The first problem is to compute frequent itemsets and the constraints are minimum support is 4 (min supp = 4) and the sum of prices>=45.
- Second problem is to find infrequent items.

In the first iteration the total price of each transaction is checked. The transactions which do not satisfy the monotone constraint (i. e.CT_M>=45) are deleted during first iteration. To count the support for the singleton items, all transaction with a sum of prices >=45 are used. After this process only the fourth transaction is deleted.

At the end of the count we find items a1, e1, f1 and h1 as infrequent. We can save these infrequent items as they discard. It should be note that, if the fourth transaction had not been deleted, items a1 and e1 would have been counted as frequent. Now on this point α-reduction is performed on the dataset that is remove a1, e1, f1 and h1 from all transactions in the dataset. After the α-reduction we have the more chances to *μ*-reduce the dataset. Note that, At the beginning, TID-2 has a total price of 63 and due to the pruning of a1 and e1 now its total price reduced to 38. This transaction i.e. TID-2 can pruned away now. The similar reasoning holds for TID-7 and TID-9. Now ExAnte counts once again to determine the support of alive items with the reduced dataset. See in Table 3, the item g1 which initially has got a support of 5 now has become infrequent. Now we can α-reduce again the dataset, and after then *μ*-reduce.

After the two reductions, TID-5 does not satisfy anymore the monotone constraint hence it is pruned away. ExAnte counts again the support of items on the reduced datasets but this time no more items are found which turned infrequent.

Finally we get the fix-point at the third iteration: the dataset has been reduced from 9 transactions to 4 transactions (number 1,3,6 and 8), and interesting itemsets have shrunk from 8 to 3 (b1, c1, and d1). At this point any constrained frequent pattern mining algorithm would find very easily the unique solution to problem which is the 3-itemset {b1,c1,d1}. But we need infrequent items. So, at the end all frequent items that are b1, c1 and d1 are discarded from total items (that are a1, b1, c1, d1, e1, f1, g1 and h1) and save all infrequent items which are a1, e1, f1, g1 and h1.

NOTE : We can also save frequent items but our main purpose of proposed algorithm to find rare items.

Table 1: Price table

Item	Price
a1	5
b1	8
c1	14
d1	30
e1	20
f1	15
g1	6
h1	12

Table 2: Transactional Database

TID	Itemset	Total price
1	b1,c1,d1,g1	58
2	a1,b1,d1,e1	63
3	b1,c1,d1,g1,h1	70
4	a1,e1,g1	31
5	c1,d1,f1,g1	65
6	a1,b1,c1,d1,e1	77
7	a1, b1,d1,f1,g1,h1	76
8	b1,c1,d1	52
9	b1,e1,f1,g1	49

TABLE 3: Item and their support (iteration)

	Support		
Items	1st	2nd	3rd
a1	3	-	-
b1	7	4	4
c1	5	5	4
d1	7	5	4
e1	3	-	-
f1	3	-	-
g1	5	3	-
h1	2	-	-

EXPERIMENT RESULT: Simulation is done in JAVA codes under NetBeans IDE and backend itemset is in the form of transactional dataset. The implementation is tested on 9 items itemset and 150 transaction itemset. The experiment results are shown in screen shot listing infrequent itemset with 99% accuracy.

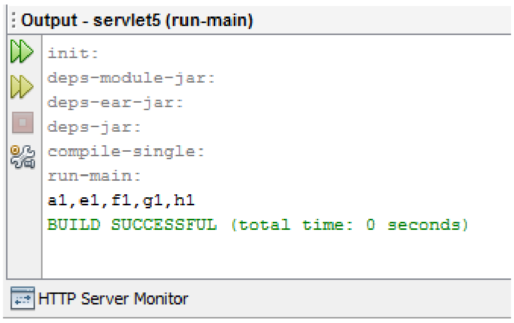
Finding of rare items

Number of items (item [0, 7]) =8

Number of transaction (itemSet [0, 8]) =9

Rare items= {a1, e1, f1, g1, h1}

All experiments are perform using NetBeans IDE 6.9 on Intel(R) Core(TM) CPU 2.10 GHz, 3GB RAM and programming is done in core java.



```
init:
deps-module-jar:
deps-ear-jar:
deps-jar:
compile-single:
run-main:
a1,e1,f1,g1,h1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Fig.- Screen shot using NetBeans IDE

- **Execution time**

Grahne et al[6], found that 80% CPU was used for traversing the FP-tree but By using ExAnte method CPU usage have reduced to 12% and also reduce execution time. In this way this algorithm is much effective to reduce search space and large input database.

V. CONCLUSION

In this paper we have introduced ExAnte, a pre-processing data reduction technique which reduces dramatically the search space and input database. And also reduces CPU usage up to 12% as compare to FP-growth* and hence execution time. We have proved experimentally the effectiveness of our algorithm using different constraint on various dataset to find infrequent itemsets. Proposed work is simulated on transactional itemset of limited size where as in future it can be implemented on real time transactional databases.

REFERENCES

[1] R. Agrawal, T. Imielinski, and Swami, “Mining Association Rules between Sets of Items in Large Databases,” Proc. *ACM SIGMOD* Int’l Conf. Management of Data (SIGMOD ’93), pp. 207-216, 1993.

[2] W. Wang, J. Yang, and P.S. Yu, “Efficient Mining of Weighted Association Rules (WAR),” Proc. Sixth ACM SIGKDD Int’l Conf. Knowledge Discovery and data Mining (KDD ’00), pp. 270-274, 2000.

[3] F. Tao, F. Murtagh, and M. Farid, “Weighted Association Rule Mining Using Weighted Support and Significance Framework,” Proc. nineth ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (KDD ’03), pp. 661-66, 2003..

[4] C.-K. Chui, B. Kao, and E. Hung, “Mining Frequent Itemsets from Uncertain Data,” Proc. 11th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD ’07), pp. 47-58, 2007.

[5] Luca Cagliero and Paolo Garza “Infrequent Weighted Itemset Mining using Frequent Pattern Growth”, IEEE Transactions on Knowledge and Data Engineering, pp. 1-14, 2014.

[6] Grahne O. and Zhu J. “Efficiently Using Prefix-trees in Mining Frequent Itemsets”, In Proc. of the IEEE ICDM Workshop on Frequent Itemset Mining, 2004.

[7] Grahne G. and Zhu J., “Efficiently Using Prefix-Trees in mining Frequent Item sets,” Proc. ICDM 2003Workshop Frequent Item set Mining Implementations, (2003).

[8] Koh, Y.S., Rountree, N.: “Finding sporadic rules using Apriori-inverse”. In: Ho, T.-B., Cheung,D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 97–106. Springer, Heidelberg (2005).

[9] Haglin DJ, Manning AM (2007) “On minimal infrequent itemset mining”.In: DMIN. CSREA Press,Las Vegas, pp 141–147.

[10] Szathmary, L., Napoli, A., Valtchev, P.: “Towards rare itemset mining”. In: Proceedings of the19th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2007, vol. 01,pp. 305–312. IEEE Computer Society, Los Alamitos (2007).

[11] Gupta, Ashish; Mittal, Akshay; Bhattacharya, Arnab,”Minimally Infrequent Itemset Mining using Pattern-Growth Paradigm and Residual Trees” *International Conference on Management of Data* (COMAD), 2011.

[12] David Huang, Yun Sing Koh, Gillian Dobbin” Rare Pattern Mining on DataStream” Data Warehousing and Knowledge Discovery Lecture Notes in Computer Science Volume 7448, 2012, pp 303-314.

[13] Tsang, S., Koh, Y.S., Dobbie, G.: RP-Tree: Rare Pattern Tree Mining Data Warehousing and Knowledge Discovery Lecture Notes in Computer Science Volume 6862, 2011, pp 277-288. Springer, Heidelberg (2011).

[14] R.Lakshmi, C.Sweetlin Hemalatha, V.Vaidehi.”Mining Infrequent Patterns in Data Stream” International Conference on Recent Trends in Information Technology 2014.

[15] R. Agrawal, R. Srikant Fast Algorithms for Mining Association Rules in Large Databases. In Proceedings of the *Twentieth International Conference on Very Large Databases*, pages 487-499, Santiago, Chile, 1994.

[16] R. T. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD-98)*, volume 27,2 of *ACM SIGMOD Record*, pages 13–24, New York, June 1–4 1998. ACM Press.

[17] J. Pei, J. Han and L. V. S. Lakshmanan. “Mining frequent itemsets with convertible constraints”. (In *Proc. of ICDE’01)*, pages 433–442, 2001.

[18] J. Han, J. pei and Y. Yin. “Mining frequent patterns without candidate generation”. In *Proc of ACM SIGMOD’00.*

[19] F. Bonchi, F. Giannott, A. Mazzanti and D. Pedreschi. “ExAnte: Anticipated Data Reduction in Constraint Pattern Mining”. In *Proc. of PKDD03.*