A Two-layer Model for Interactive Mining of Frequent Patterns

M.H Nadimi-Shahraki, *Member, IEEE*, Norwati Mustapha

***Abstract*—Commonly, frequent patterns are mined by satisfying a user specified minimum support threshold or minsup in short. In some applications, finding proper frequent patterns by changing the value of minsup is needed. Since rerunning the mining algorithm from scratch can be very time consuming and may unacceptable for real time applications, researchers have introduced interactive mining. Although need to interactive mining has been addressed in many studies, thus far there has not been proposed any specified model to develop an efficient interactive mining method. In this paper, we propose a two-layer model including mining model construction and mining process for interactive mining. The experimental results verify that using the proposed model avoids database rescanning and reconstructing of the mining model which is the basic idea to enhance the efficiency of interactive mining.**

***Index Terms*—Frequent pattern mining, interactive mining, mining model.**

I. Introduction

THE explosive growth of data in all business, government and scientific applications creates enormous hidden knowledge in their databases. Certainly, in this decade knowledge discovery or extracting knowledge from large amount of data is a desirable task in competitive businesses. Knowledge discovery from databases (KDD) is an interactive and iterative process and usually it starts from raw data to mine finally proper data patterns by data mining tasks. Therefore, data mining is an essential step in process of KDD. Since the introduction of the Apriori algorithms [2], frequent patterns mining plays an important role in data mining tasks such as clustering, classification, prediction and association analysis. Frequent patterns are itemsets that exist in a dataset with frequency no less than a user specified minimum support threshold or minsup in short.

The past decade has seen the rapid development and diffusion of several approaches to mine frequent patterns more efficiently. They are almost based on three fundamental frequent patterns mining methodologies: Apriori, FP-tree and Eclat [10]. Commonly, frequent patterns are mined by satisfying a specified minsup, and an appropriate value for minsup can reduce the time and space costs of frequent pattern mining. However, it is not easy to find an appropriate value of minsup, because the appropriateness depends on the application and the expectation of the user [3, 7, 16]. Therefore, there is a need to rerun the mining algorithm on the same (relevant) data by various minsup to find proper frequent patterns. Since rerunning the mining algorithm from scratch can be very time consuming and costly, researchers have introduced interactive mining where the transaction database remains unchanged and only the minsup is changed to find proper frequent patterns.

Unfortunately, both Apriori [2] and Eclat [17] approaches cannot be easily adoptable with interactive mining. Thus far a few efficient interactive mining methods [5, 11, 13, 14] have been introduced mostly based on FP-tree approach [9]. An important result gained by analyzing the efficient works is that, avoiding database rescanning and rebuilding of the mining material is the basic idea to implement an efficient interactive mining method. In other words, an efficient interactive mining method must fit “build once, mine many” principle [5, 6] such that once the content is captured, then it can be frequently mined with various values of minsup.

In this paper, we propose a two-layer model to develop an efficient interactive mining method. It proposes construction of the mining model in the first layer isolated from the mining process considered in the second layer. The experimental results verify that by using the proposed model no need to database rescan and reconstructing of the mining model which is the basic idea to develop an efficient interactive mining method.

II. Problem and Related Work

A. Problem Description

Frequent patterns are itemsets or substructures that exist in a dataset with frequency no less than a user specified

Dr. M.H Nadimi-Shahraki is with faculty of computer engineering, Islamic Azad University, Najafabad branch (IAUN), Iran, email: nadimi@ieee.org.
Dr. Norwati Mustapha is with faculty of computer science and information technology, University of Putra Malaysia (UPM), Selangor, Malaysia, email: norwati@fsktm.upm.edu.my.

threshold called minsup. Let *L*= {i_1, i_2 … i_n} be a set of items. Let *D* be a set of database transactions where each transaction *T* is a set of items and |*D*| be the number of transactions in *D*. Given *P*= {i_j … i_k} be a subset of *L* ($j \leq k$ and $1 \leq j, k \leq n$) is called a pattern. The support of pattern *P* or S (*P*) in *D* is the number of transactions in *D* that contains *P*. The pattern *P* will be called frequent if its support is no less than a user specified support threshold minsup σ ($0 \leq \sigma \leq |D|$). The problem of frequent pattern mining is finding all frequent patterns (FP) in *D* with respect to σ denoted by FP (σ).

In some real time applications such as web usage mining and online recommendation systems, finding new correlations between items by changing minsup is very useful. When users change minsup, finding frequent patterns with respect to new minsup in an acceptable response time is expected. Avoiding database rescanning and updating of frequent pattern model is the basic idea to implement an efficient algorithm. Let *σ'* be new minsup then the problem of interactive mining of frequent patterns is to find all frequent patterns in *D* with respect to new minsup *σ'* or FP (σ') using the current mining model. It means in interactive mining the content must be captured once and then it can be frequently mined by various values of minsup. It has been called "build once, mine many" principle [5, 11, 13].

Although need to interactive mining has been addressed in many studies [1, 5, 6, 8, 11, 13, 15], thus far there has not been proposed any specified model to develop an efficient interactive mining method. In this paper, we propose a two-layer model for interactive mining of frequent patterns.

B. Related Work

Since the introduction of the Apriori algorithms [2], frequent patterns mining plays an important role in data mining tasks such as clustering, classification, prediction and association analysis. Many efficient algorithms have been introduced to solve the problem of frequent pattern mining more efficiently. They are almost based on three fundamental frequent patterns mining methodologies: Apriori, FP-growth and Eclat [10]. Unfortunately, both Apriori and Eclat approaches cannot be easily adoptable with interactive mining. The main reason is that, they cannot fit “build once, mine many” principle whereby when the minsup is changed; the mining algorithm must be started from scratch. On the other hand, the FP-tree approach needs to scan database twice to keep only frequent items in memory. When minsup is changed then the FP-tree is reconstructed by two database scan with respect to new minsup. Although the original FP-tree has this kind of weakness, it has potential to be extended for interactive mining. Thus far a few efficient interactive mining methods have been introduced which fit "build once, mine many" principle. Mostly, they have improved the FP-tree to capture the content of relevant data by one database scan. They construct the tree once independent of minsup and then usually use the original FP-growth to explore the tree with various minsup. As a good result, when minsup is changed, there is no need to rerun the algorithm from scratch. Their performance study shows that they are efficiently applicable for interactive mining of frequent patterns.

Cheung and Zaiane [5] proposed the CATS tree and FELINE algorithm mainly for interactive mining. The CATS tree is an extension of the FP-tree to improve storage compression. The aim is to build a prefix tree as compact as possible by only one database scan. It adds new transactions at the root level and then compares their items with children (or descendant) nodes, which arranged in descending order. If in both the new transaction and the children nodes same items exist, then it merges the transaction with the node at the highest frequency level, and the remainder of the transaction is added to the merged nodes. Recursively, this process is repeated until all common items are found. If the node’s frequency becomes higher than the frequencies of its ancestors, then it will be swapped with the ancestors to keep the descending local ordering. Their experiment results showed that CATS Tree and the FELINE algorithm were well-suited for interactive mining. But, it still needs lots of swapping, merging, and splitting of tree nodes, because items in the trees are arranged base on a global frequency-dependent ordering.

AFPIM algorithm [11] introduces adjusting FP-tree which is an extension of FP-tree using the original notion of FP-tree. Similar FP-tree, it keeps only frequent items but respect to a threshold called preminsup, which is reasonably lower than minsup. Consistently, deletions, insertions and/or modifications may affect the frequency of items, which affects the ordering of items. It results in adjusting items in the tree. The AFPIM swaps such items by using bubble sort to exchange adjacent items recursively. The bubble sort needs to test all the branches affected by the item frequency ordering, therefore the swapping operation can be very expensive. Moreover, determining an appropriate preminsup is difficult.

Leung et al. [12, 13] proposed a canonical-order tree or CanTree in short. It is an efficient extension of the FP-tree to capture the content of the transaction database by one database scan in canonical order, specifically; items can be consistently arranged in lexicographic order. Hence, there is no need to search and find merge-able paths like those in the CATS tree, and swapping of tree nodes affected by the frequency ordering. Once the CanTree is constructed, frequent patterns can be mined from the tree by using the original FP-growth. The simplicity and less cost of CanTree construction solve the weaknesses of the CATS and AFPIM. However, its compactness is not similar to the FP-tree and CATS tree, especially when datasets are sparse, probability for tree nodes to share common paths is drop and the compactness rate of CanTree is decreased and tree will be wide. Consequently, the mining process takes more time.

We previously proposed [14] a prime-base and compressed tree or PC-tree in short. The content of relevant data is captured by PC-tree with one database scan and mining materials are consequently formed. The PC-tree is a well-organized tree structure, which is systematically built based on descendant making. Moreover, this study introduces a mining algorithm called PC-miner to mine the mining model frequently with various values of minsup. It grows an effective candidate head starting from the longest candidate patterns by using the Apriori principle. Meanwhile, during the growing of the candidate head set in each round, the longest candidate patterns are used to find maximal frequent patterns from which the frequent patterns can be derived. Moreover, the PC-miner reduces the number of candidate patterns and comparisons by using several pruning techniques. Their experimental results showed that their method fits "build once, mine many" principle and it is efficient for interactive mining.

III. Two-layer Model for Interactive Mining

As explained in previous section, the need to interactive mining has been addressed in many studies [1, 5, 6, 8, 11, 13, 15]. However, they did not propose any specified model for interactive mining method. An important result gained from analyzing efficient interactive mining methods is that, the avoiding database rescanning and rebuilding of the mining material is the basic idea to develop an efficient interactive mining algorithm. In other words, an efficient interactive mining method must fit the “build once, mine many” principle, where the mining model is constructed once, and then it can be used by mining process with various minsup values. According to the above discussion, we define two main components: mining model and mining process as follows.

Mining model: Consider relevant database *DB* and mining method *M* to mine frequent patterns of *DB*. Let *S*= (S_1 … S_j) be a set of different tasks which must be done by *M* to mine all frequent patterns of *DB*. The mining model construction consists of doing all possible tasks that are independent of the minimum support threshold minsup. Moreover, the results formed by constructing mining model which can be used to mine frequent patterns is called mining materials.

Mining process: The mining process consists of the rest of tasks in *S* which dependent on the minsup. It mines frequent patterns by using the mining materials formed in the mining model construction.

Based on the above discussions, in this section we propose a two-layer model for interactive mining of frequent patterns. As shown in Figure 1, the proposed model consists of two isolated layers: mining model construction and mining process. Obviously, in interactive mining, rerunning the mining algorithm from scratch is very time consuming, and it results in an unacceptable response time. Hence, in the proposed model, the mining model is constructed independent of minsup. That is because, once the mining materials are made, they can be frequently used by mining process with respect to various minsup. In fact, in the proposed model, the mining model is isolated from the mining process. Usually, before starting the mining process, several steps must be done to make mining materials from which frequent patterns can be mined. Consistently, the mining model construction is started by scanning the relevant data, but it is very important that the mining model construction can be consisted of more than only database scanning.

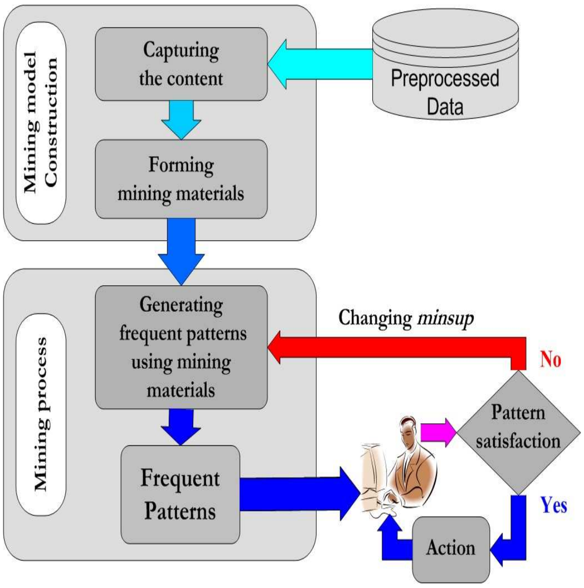

Figure 1. Two-layer model for interactive mining.

Moreover, we aim to avoid database rescanning when minsup is changed. Therefore, in the mining model construction the content of relevant data should be captured. Capturing the content usually consists of database scanning and keeping the content in memory by a well-organized structure. Reasonably, forming the mining materials that are more informative can enhance the efficiency of interactive mining. Usually, it results in increasing the time and computational costs required for constructing the mining model. Although the above approach increases the cost of the mining model construction, the cost of such expensive mining model will be amortized over the mining model’s estimated life in several runs of mining process.

IV. Experimental Results

This section is to show that the proposed model fits "build once, mine many" principle by which database rescanning and reconstructing of mining model are avoided in interactive mining. As explained in previous section, when the mining model is constructed independent of minsup, then it can be frequently mined by the mining process with various minsup. Since both CanTree and PC-tree are adoptable with the proposed model, they are experimentally compared with FP-tree. For doing this, all algorithms have been implemented in C. Moreover, all experiments are run on windows-XP with a 2.4 GHz CPU and 2 GB memory. In each experiment, the algorithms are separately run in the same experimental environment. The results reported in this section have been computed by the average of multiple runs.

The experimental evaluation is conducted by the popular synthetic dataset T10I4D100K and real dataset mushroom. T10I4D100k is generated by the program developed at IBM Almaden Research Center [2]. The number of transactions, the average transaction length and the average frequent pattern length of T10I4D100k are set to 100k, 10 and 4 respectively. The real dataset mushroom is downloaded from UC Irvine Machine Learning Depository [4]. The records of mushroom consist of the characteristics of various mushroom species, and the number of records, the number of items and the average record length are set to 8124, 119 and 23 respectively.

In the first experiment, the total time of mining model construction for static mining is evaluated. Since both PC-tree and CanTree may increase the cost of the mining model construction, it is expected that the cost of the FP-tree is less than PC-tree and CanTree. The bar charts shown in Figure 2 agree with the expectation. Although the mining model construction in PC-tree and CanTree is independent of minsup, in this experiment the minsup is set to 0.1% and 20% for T10I4D100K and mushroom respectively. This experimental results show that the FP-tree is efficient than PC-tree and CanTree for static mining.

In the second experiment, the total time of mining model construction for interactive mining is evaluated. There are two kinds of scenarios to change the minsup, when the value of new minsup is higher and/or lower than the old one. Since in the first kind scenarios, the interactive mining methods can find frequent patterns satisfying the new minsup by using cached frequent patterns. Therefore, in the second experiment, we only examine the cost of mining model construction for interactive mining in the second kind scenarios when the new minsup is lower than the old one. It is called descending scenario which consists of a descending sequence of minsup. Although this experiment was run over several descending sequences, according to the space limitation, the experimental result of only one descending sequence for each dataset is presented. The descending sequence for synthetic dataset T10I4D100K is started with minsup value of 2%, and then it is decreased to 1% to find more patterns that are frequent. Then, minsup is decreased to 0.5% and finally to 0.1% where has been considered that the proper frequent patterns are found. Moreover, the descending sequence for real dataset mushroom consists of minsup with values of 30%, 25%, 20% and 15%.

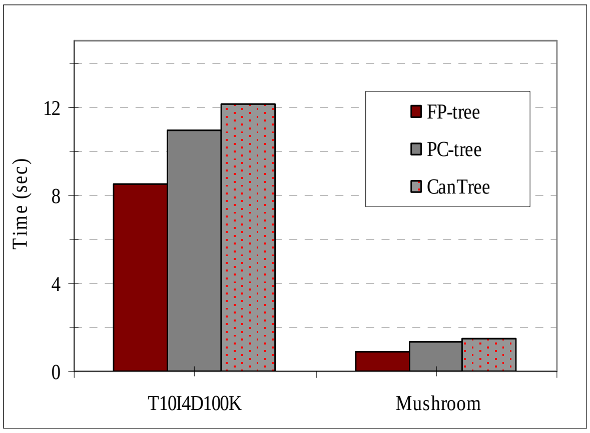

Figure 2. Time of mining model construction.

The graphs plotted in Figure 3 and 4 show the total runtime required for constructing the mining model by different methods PC-tree, CanTree and FP-tree over the above descending scenarios. Since both methods PC-tree and CanTree construct the mining model independent of minsup, they fit "build once, mine many" principle and there is no need to reconstruct the mining model when minsup is changed.

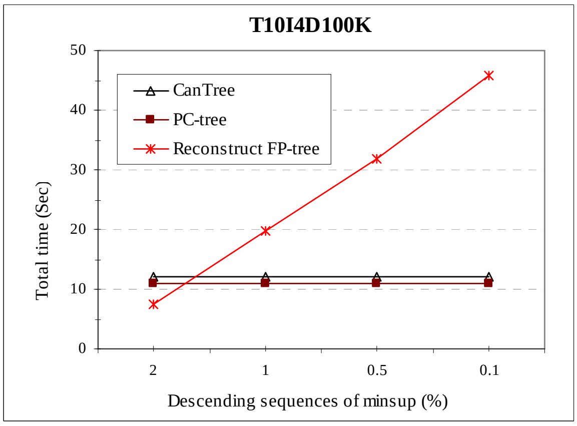

Figure 3. Total time of mining model construction for interactive mining over synthetic dataset T10I4D100K.

Therefore, the cost of mining model construction for new minsup is equal to zero and total runtime required for constructing the mining model is fixed and identical to the spent for the first run. Conversely, in interactive mining, the FP-tree must be reconstructed and the number of changes in minsup increases the total runtime of mining model construction which results in increasing the total response time of interactive mining.

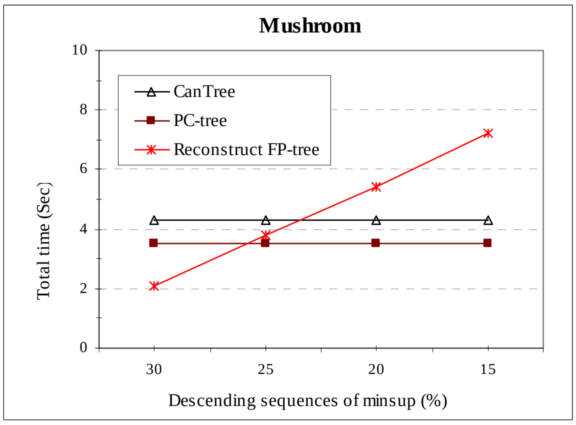

Figure 4. Total time of mining model construction for interactive mining over real dataset mushroom.

V. Conclusions and Future Work

Researchers have introduced interactive mining to avoid rerunning the mining algorithm from scratch. Although need to interactive mining has been addressed in many studies [1, 5, 6, 8, 11, 13, 15], there has not been proposed any specified model for developing an efficient interactive mining method. In this paper, we propose a two-layer model for interactive mining of frequent patterns. The first layer is to construct the mining model independent of minsup. Consequently, the second layer consists of mining process. The cost of FP-tree was experimentally compared with interactive mining methods PC-tree and CanTree which are adoptable with the proposed model.

The experimental results verified that by using the proposed model, the mining model is constructed once, and when minsup is changed there is no need to database rescanning and reconstructing the mining model. Although the cost of FP-tree for static mining was less than both PC-tree and CanTree, in interactive mining, the FP-tree must be reconstructed and the number of changes in minsup increases its cost. Conversely, by using the proposed model, both PC-tree and CanTree construct the mining model only in first run in interactive mining which results to reduce the response time for interactive mining. The proposed model can be also used to develop further superior interactive mining methods.

References

[1]. Aggarwal, C.C. and P.S. Yu, "A new approach to online generation of association rules", IEEE Transactions on Knowledge and Data Engineering, 2001. Vol. **13**(4): p. 527-540.

[2]. Agrawal, R. and R. Srikant, "Fast algorithms for mining association rules", International Conference on Very Large Data Bases (VLDB’94), 1994: p. 487-499.

[3]. Bing, L., et al., "Finding interesting patterns using user expectations", IEEE Transactions on Knowledge and Data Engineering, 1999. Vol. **11**(6): p. 817-832.

[4]. Blake, C. and C. Merz, "UCI repository of machine learning databases. University of California – Irvine, Irvine, CA", 1998.

[5]. Cheung, W. and O.R. Zaiane, "Incremental mining of frequent patterns without candidate generation or support constraint", Seventh IEEE International Database Engineering and Applications Symposium (IDEAS’03), 2003: p. 111-116.

[6]. El-Hajj, M. and O. Zaiane, "Inverted matrix: Efficient discovery of frequent items in large datasets in the context of interactive mining", the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003: p. 109-118.

[7]. Geng, L. and H. Hamilton, "Interestingness measures for data mining: A survey", ACM Computing Surveys (CSUR), 2006. Vol. **38**(3): p. 1-32.

[8]. Goethals, B. and J. Van den Bussche, On Supporting Interactive Association Rule Mining, Data Warehousing and Knowledge Discovery, 2000, p. 307-316.

[9]. Han, J., J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation", 2000 ACM SIGMOD Intl.Conference on Management of Data, 2000: p. 1–12.

[10]. Han, J., et al., "Frequent pattern mining: current status and future directions", Data Mining and Knowledge Discovery, 2007. Vol. **15**(1): p. 55-86.

[11]. Koh, J.L. and S.F. Shieh, "An efficient approach for maintaining association rules based on adjusting FP-Tree structures", Lecture Notes in Computer Science, 2004: p. 417-424.

[12]. Leung, C.K.S., Q.I. Khan, and T. Hoque, "CanTree: a tree structure for efficient incremental mining of frequent patterns", Proc. ICDM 2005, 2005: p. 274–281.

[13]. Leung, C.K.S., et al., "CanTree: a canonical-order tree for incremental frequent-pattern mining", Knowledge and Information Systems, 2007. Vol. **11**(3): p. 287-311.

[14]. Nadimi-Shahraki, M., et al. "A New Method for Interactive Mining of Frequent Patterns", 5th International Data Mining Conference (DMIN’09), USA, 2009

[15]. Parthasarathy, S. and S. Dwarkadas, "Shared State for Distributed Interactive Data Mining Applications", Distributed and Parallel Databases, 2002. Vol. **11**(2): p. 129-155.

[16]. Silberschatz, A. and A. Tuzhilin, "What makes patterns interesting in knowledge discovery systems", Knowledge and Data Engineering, IEEE Transactions on, 1996. Vol. **8**(6): p. 970-974.

[17]. Zaki, M.J., "Scalable algorithms for association mining", IEEE Transactions on Knowledge and Data Engineering, 2000. Vol. **12**(3): p. 372-390.