An Incremental Mining Algorithm for Erasable Itemsets

Tzung-Pei Hong
Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, Taiwan
tphong@nuk.edu.tw

Kun-Yi Lin
Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, Taiwan
liquid-8@hotmail.com

Chun-Wei Lin
School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China
jerrylin@ieee.org

Bay Vo
Faculty of Information Technology, Ho Chi Minh City University of Technology,
Ho Chi Minh, Viet Nam
bayvodinh@gmail.com

***Abstract*—Erasable-itemset (EI) mining is to find the itemsets that can be eliminated but do not greatly affect the factory's profit. In this paper, an incremental mining algorithm for erasable itemset is proposed. It is based on the concept of the fast-update (FUP) approach, which was originally designed for association mining. Experimental results show that the proposed algorithm executes faster than the batch approach in the intermittent data environment.**

Keywords—data mining, erasable-itemset mining, FUP algorithm, incremental mining

I. INTRODUCTION

Traditionally, the most popular data mining considers itemset frequencies to get useful association rules [1][2][3][9]. In 2009, Deng et al. defined the problem of erasable-itemset (EI) mining, which is a variant of pattern mining [5]. It originates from production planning associated with a factory that produces many different types of products from a variety of materials. Each product is created from several items (materials) and has its own profit. In order to produce all the products, the factory has to buy and store these different kinds of items. When considering the financial costs, the factory cannot afford to buy all the necessary items as usual. Therefore, the managers should consider their production plans to balance the profit and cost of the factory. According to the above situation, the problem is to find the itemsets that can be eliminated but do not greatly affect the factory's profit, allowing managers to create a benefit-cost trade-off production plan [5]. Deng et al. also proposed several approaches to solve the problem [5][6][7][8]. Some other scholars then improve them in efficiency by using some useful data structures [10][11][12][14].

In this paper, an incremental mining algorithm to update the discovered erasable itemsets is proposed. It is based on the FUP concept [4]. The proposed approach first partitions itemsets into four parts according to whether they are erasable in the original database and in the new products. Each part is then executed by its own procedure. The difference between the proposed algorithm and the FUP algorithm is that the itemsets to be handled are different. We use the promising candidates to compare the items appearing in the original product database and newly insert product database. Experimental results show that the proposed algorithm has a better performance than the conventional batch-mining algorithms.

II. REVIEW OF RELATED WORK

A. Erasable-Itemset Mining

Erasable-itemset mining was proposed by Deng et al. in 2009 for analyzing production planning [5]. It has some real application values for factory manufacturing. Formally, let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of m items, which represents the union of the components for different kinds of products in a factory. A product dataset, DB, contains a set P of n manufactured products, $\{p_1, p_2, \ldots, p_n\}$. Each product p_i is represented in the form of $\langle items_i, val_i \rangle$, where $items_i$ is a subset of I that constitutes p_i and val_i is the profit that the factory obtains by producing p_i. A set $X \subseteq I$ is called an erasable itemset if the sum of the profit values of the products with at least one item in X is equal to or less than a given threshold of the total profits of all products.

B. The FUP Algorithm

In real-world applications, new data are always generated intermittently. When they are inserted into databases, they will affect final mined results from old databases [13]. Some old knowledge may become invalid, or new knowledge may be derived due to the inserted data. An intuitional approach is to re-mine the knowledge from the entire updated database. It can guarantee getting correct updated knowledge, but the computational cost is high especially when new data comes in a small amount each time.

Cheung et al. thus proposed the Fast UPdate (FUP) algorithm to effectively update discovered frequent itemsets for inserted data [4]. They divided the itemsets in an original database as two cases: frequent and infrequent. Frequent itemsets were desired and stored. They also divided the items in inserted transactions as frequent and infrequent according to their counts only in the new transactions. Four cases thus exist by combining the above divisions. Each item will lie in one of

978-1-5090-5795-5/17/$31.00 ©2017 IEEE

the four cases, and is processed according to the case type. The FUP algorithm can reduce some mining time for new data, but it still needs to scan an old database if an item is infrequent in the original database but frequent in the new transactions.

III. An Incremental Algorithm for Mining Erasable Itemsets

In this section, we will present the proposed approach for incrementally mining erasable itemsets.

A. Main Idea

A factory always needs to design new products for keeping its competition especially for the businesses that emphasize small-amount but versatile production. When new products are inserted into a product database, the previous batch approach can be used to re-mine the updated erasable itemsets, but it does not utilize the knowledge already obtained so far, thus wasting execution time. In this paper, we adopt the concept of FUP in the erasable-itemset mining to save the processing time for intermittent coming of new products. Similar to FUP for association rules, we can divide the cases of incremental erasable-itemset mining into the following four cases (illustrated in Figure 1). The theoretic foundation behind Figure 1 is the weighted average gain ratio of the items in the old product database and in the new coming products.

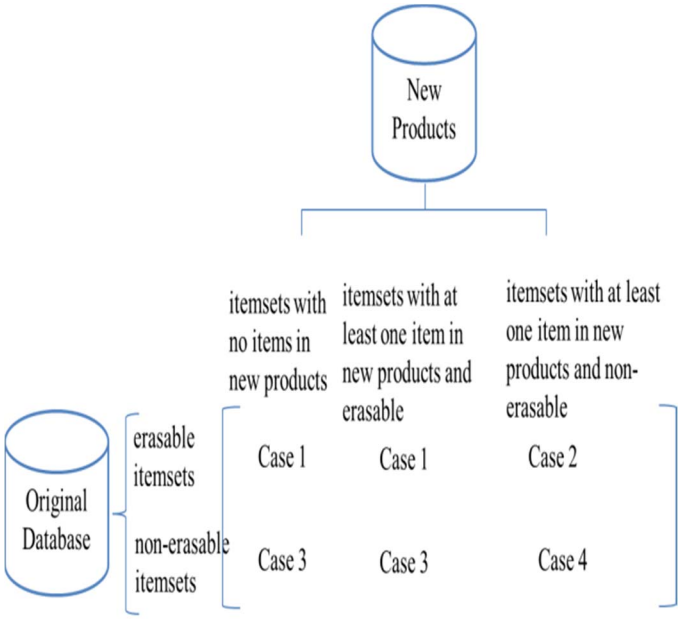

Figure 1: The four cases for modifying erasable itemsets

The itemsets in Case 1 are divided into two parts to be processed. The first part includes the itemsets which are erasable in the original product database but with no items in the new products. They will still be erasable after the process. The second part includes the itemsets which are erasable in both the original product database and the new product database. They will still be erasable after the process. Similarly, itemsets in Case 4 will still be non-erasable after the new product database is inserted. Thus Cases 1 and 4 will not affect the final erasable itemsets.

Case 2 may remove existing erasable itemsets, and Case 3 may add new erasable itemsets. Based on the FUP approach, Cases 1, 2 and 4 are more efficiently handled than conventional batch mining algorithms.

B. The Proposed Incremental Erasable Itemset Mining Algorithm

As mentioned above, when a set of new products come, the proposed algorithm will re-calculate the gain of an erasable itemset or a non-erasable itemset according to the four cases, and decide whether the itemset is erasable. The proposed algorithm is stated in details as follows.

The proposed algorithm:

Input: An original product database P with its total profit value T^P, the set of erasable itemsets E^P with their gain values from P, an erasable ratio threshold r, the set of all items I, and a set of newly coming products N.

Output: The set of erasable itemsets for the updated database U.

Step 1: Calculate the total profit value T^N of the new products N as follows:

$$T^N = \sum_{p_i \in N} p_i.value$$

where p_i is a product in N.

Step 2: List the 1-itemsets appearing in the new products N as the candidate erasable 1-itemsets C_1^N and calculate their gain values from the new products N.

Step 3: Set $C_1^P = I - C_1^N$, where C_1^P records the candidate 1-itemsets not appearing in new products.

Step 4: Set $k = 1$, where k records the number of items in the itemsets currently being processed.

Step 5: Put a candidate erasable k-itemsets in C_k^N into the set (E_k^N) of erasable k-itemsets for the new products N if its gain value in the new products is smaller than or equal to the maximum gain threshold $T^N \times r$.

Step 6: For each k-itemset s in the set of erasable k-itemsets (E_k^P) from the original product database P, if it is also in E_k^N, do the following substeps (case 1):

Substep 6-1: Set the updated gain of s as:

$$gain^U(s) = gain^P(s) + gain^N(s),$$

where $gain^U(s)$, $gain^P(s)$, $gain^N(s)$ are the gains of s in the updated product dataset, the original product dataset, and the set of new products, respectively.

Substep 6-2: Directly put s in the set of updated erasable k-itemsets, E_k^U.

Step 7: For each k-itemset s in the set of erasable k-itemsets (E_k^P) from the original product database P, if it is also in the candidate erasable k-itemsets C_k^N but not in E_k^N (i.e. $C_k^N - E_k^N$), do the following substeps (case 2):

Substep 7-1: Set the updated gain of s as:

$$gain^U(s) = gain^P(s) + gain^N(s).$$

Substep 7-2: Check whether the updated gain of s is smaller than or equal to maximum gain threshold $(T^P + T^N) \times r$. If s satisfies the condition, put it in the set of updated erasable k-itemsets, E_k^U.

Step 8: For the other k-itemsets in the set of erasable k-itemsets (E_k^P) from the original product database P, directly keep them in the set of updated erasable k-itemsets, E_k^U, with their gain values unchanged.

Step 9: For each k-itemset s which exists in (E_k^N-E_k^P) (erasable for new products, but not erasable for the original product database) or in C_k^P - E_k^P, do the following substeps (case 3):

Substep 9-1: Rescan the original product database to determine $gain^P(s)$.

Substep 9-2: If s is in (E_k^N-E_k^P), set the updated gain of s as:

$$gain^U(s) = gain^P(s) + gain^N(s);$$

Otherwise, directly set the updated gain of s as:

$$gain^U(s) = gain^P(s).$$

Substep 9-3: Check whether the updated gain of s is smaller than or equal to maximum gain threshold $(T^P + T^N) \times r$. If s satisfies the condition, put it in the set of updated erasable k-itemsets, E_k^U; Otherwise, remove s from the set of erasable k-itemsets (E_k^N) from the new products.

Step 10: Form the candidate erasable $(k+1)$-itemsets C_{k+1}^U from the k-itemsets in E_k^U in a way similar to that in the two-phase batch algorithm. If a candidate erasable $(k+1)$-itemset s generated includes at least one item in the new products, put s in C_{k+1}^N and calculate its gain value from the new products N. else put s in C_{k+1}^P.

Step 11: Set $k = k+1$.

Step 12: Repeat Steps 5-11 until no new candidate erasable itemsets are generated.

Step 13: Output all the updated erasable itemsets generated so far.

IV. Experiments and Analysis

In this section, we compare the performance of both the proposed incremental algorithm (called FUP-erasable) and the META-erasable algorithm [5] which was executed in a batch way. Several simulation datasets were generated and used in the experiments to evaluate the performance.

The first 100000 products were generated to serve as the original product database. The number of items contained in each product was then randomly decided from 1 to L (40 in this case). Then the items were randomly selected from the pool of total items (the number is I). Next, the gain value of each product was randomly generated in the range from 100 to the maximum gain value (3000 in this case). The original product database was then used to derive the initial set of erasable itemsets. In the experiments, each time 1% (1000) of the total number of products was then generated sequentially as inserted products. The maximum threshold was from 5% to 65% to evaluate the performance of the proposed incremental algorithm (FUP-erasable) and the META-erasable algorithm. Figure 2 shows the execution time required by the two algorithms for processing 1 insertion (with 1% (1000) new products).

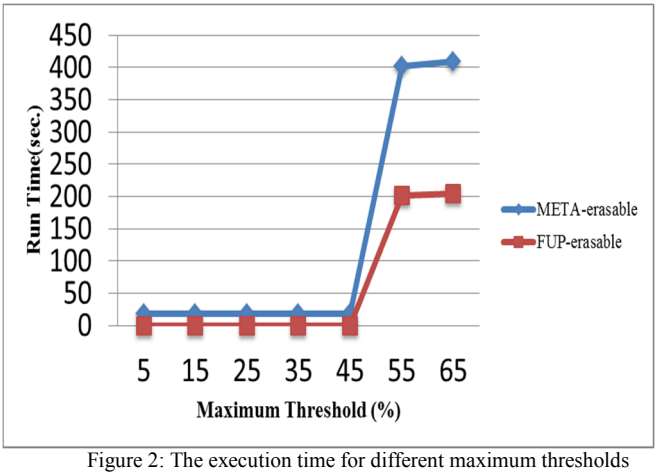

Figure 2: The execution time for different maximum thresholds

The results showed the proposed incremental FUP-erasable algorithm had better performance than the META-erasable algorithm. Since a higher maximum threshold can cause more erasable itemsets, the run time in the different maximum thresholds 55% and 65% is much higher than that in 5% to 45%. The numbers of erasable itemsets by both the two approaches are shown in Figure 3.

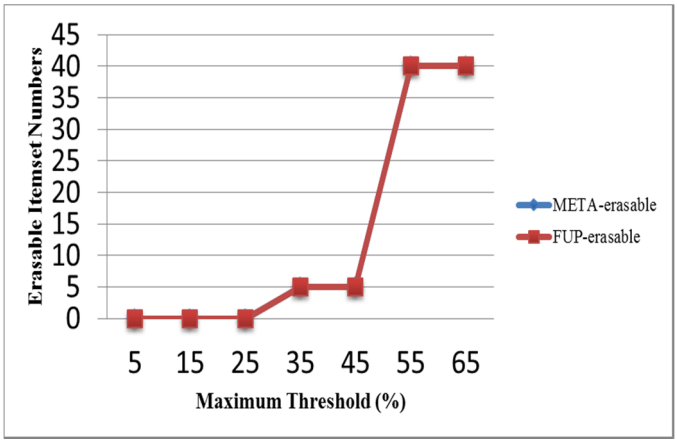

Figure 3: The numbers of erasable itemsets for different maximum thresholds

In Figure 3, the numbers of erasable itemsets for the two methods are exactly the same for each maximum threshold, proving the correctness of the proposed approach.

V. Conclusions and Future Work

In this paper, an incremental algorithm for efficiently mining erasable itemsets is proposed for product insertions based on the FUP concept. Based on the proposed approach, it can thus re-use the already discovered information to efficiently maintain and update the erasable itemsets without re-scanning the entire updated database for handling inserted products. The experimental results also show that the performance of the proposed algorithm executes faster than

the META-erasable algorithm in the intermittent data environment in incremental mining. From the numbers of erasable itemsets in the experiments, we may know the proposed approach can get the same mining results as re-running the batch algorithm. The proposed approach is thus indeed feasible in the incremental situation. In the future, we will extend the proposed approach to more complicated mining tasks.

ACKNOWLEDGMENT

This research was supported by Ministry of Science and Technology, Taiwan, R.O.C. under grant no. MOST 105-2221-E-390-017-MY2.

REFERENCES

[1] R. Agrawal, T. Imielinski and A. Swami, "Database mining: a performance perspective," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, pp. 914-925, 1993.

[2] R. Agrawal, T. Imielinski and A. Swami, "Mining association rules between sets of items in large databases," *The 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207-216, 1993.

[3] R. Agrawal and R. Srikant, "Fast algorithm for mining association rules," *The 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994.

[4] D. W. Cheung, J. Han, V. T. Ng, and C. Y. Wong, "Maintenance of discovered association rules in large databases: An incremental updating approach," *The Twelfth International Conference on Data Engineering*, pp. 106-114, 1996.

[5] Z. H. Deng, G. Fang, Z. Wang and X. Xu. "Mining erasable itemsets," *The 2009 International Conference on Machine Learning and Cybernetics*, Vol. 1, pp. 67–73, 2009.

[6] Z. H. Deng and X. R. Xu, "An efficient algorithm for mining erasable itemsets" *The Sixth International Conference on Advanced Data Mining and Applications*, pp. 214–225, 2010.

[7] Z. H. Deng and X. R. Xu, "Fast mining erasable itemsets using NC_sets," *Expert Systems with Applications*, Vol. 39, Issue 4, pp. 4453–4463, 2012.

[8] Z. H. Deng, "Mining top-rank-k erasable itemsets by PID_lists," *International Journal of Intelligent Systems,* Vol. 28, Issue 4, pp. 366-379, 2013.

[9] J. Han, J. Pei, Y. Yin and R. Mao, "Mining frequent patterns without candidate generation: a frequent-pattern tree approach," *Data Mining and Knowledge Discovery*, Vol. 8, Issue 1, pp. 53-87, 2004.

[10] T. Le, B. Vo and F. Coenen, "An efficient algorithm for mining erasable itemsets using the difference of NC-Sets" *The 2013 International Conference on Systems, Man, and Cybernetics*, pp. 2270–2274, 2013.

[11] T. Le and B. Vo, "MEI: an efficient algorithm for mining erasable itemsets" *Engineering Applications of Artificial Intelligence*, Vol. 27, pp.155–166, 2014.

[12] T. Le, B. Vo and G. Nguyen. "A survey of erasable itemset mining algorithms," *Data Mining and Knowledge Discovery*, Vol. 4, Issue 5, pp. 356-379, 2014.

[13] B. Nath, D. K. Bhattacharyya and A. Ghosh, "Incremental association rule mining: a survey," *Data Mining and Knowledge Discovery*, Vol. 3, Issue 3, pp. 157-169, 2013.

[14] G. Nguyen, T. Le, B. Vo and B. Le, "A new approach for mining top-rank-k erasable itemsets," *The Sixth Conference on Intelligent Information and Database Systems,* pp. 73–82, 2014.