Mining Erasable Itemsets with Multiple Thresholds under the Loose Constraint

Tzung-Pei Hong[1,2], Yi-Chen Chang[2], Chun-Ho Wang[2], and Wei-Ming Huang[3]
[1]Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, Taiwan
[2]Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan
[3]Department of Electrical and Control, China Steel, Inc., Kaohsiung, 806, Taiwan
Email: tphong@nuk.edu.tw, 4a0g0902@stust.edu.tw, dodo166577@gmail.com, granthill168@gmail.com

***Abstract*—In data mining, erasable-itemset mining is a popular research field widely applied in factory production management. Traditional erasable-mining algorithms typically use a single threshold as the criterion for mining erasable itemsets. It implicitly assumes that all items are of equal importance, neglecting the fact that each item has a distinct value in an application. In this paper, we use the concept of multiple thresholds and employ the loose constraint to calculate the threshold of an itemset. Since the downward-closure property is not applicable to the loose constraint, we thus utilize the sorted closure to narrow the search space and improve search efficiency. Through experiments, we compare the performance of the erasable-itemset mining using single and multiple thresholds.**

Keywords-data mining; erasable-itemset mining; multiple thresholds, loose constraint.

I. INTRODUCTION

In the digital generation, we are confronted with vast amounts of data containing useful information and potential value. These massive datasets hide important patterns, trends, and knowledge. It is a challenging task to extract these treasures efficiently from the data. Data mining, an interdisciplinary technology, has thus been proposed to reveal latent information within data and help us catch the implicit meaning embedded in the data.

Data mining is a technology that combines statistics, mathematics, databases, and artificial intelligence. Its goal is to discover patterns, trends, and knowledge within big data. By applying various data analysis techniques, data mining can uncover previously unknown patterns, providing robust support for decision-making, prediction, and optimization. The applications of data mining are extensive, spanning areas such as business, healthcare, finance, social sciences, environmental sciences, and so on. In the business domain, data mining is widely used in market analysis [2][4][5], customer relationship management, risk management, and other aspects to enhance the competitiveness of a company.

Erasable itemset mining [8] is one of the essential mining problems with extensive applications. It deals with the problem that raw materials for making products cannot be entirely purchased in a factory due to some unforeseen problems encountered suddenly, such as insufficient funds, limited logistics transportation capacity, insufficient storage space, etc. Therefore, it is necessary to make a decision for choosing which raw materials should not be purchased so that the company's loss can be controlled within an acceptable range.

In traditional erasable-itemset mining, an item (raw material) or an itemset (a set of raw materials) is called erasable if the profit-loss ratio of not generating the products that need to use the materials is larger than or equal to a single threshold. In this case, each item (raw material) is treated as equally important. However, it is sometimes unfair and impractical because other properties of the materials are not considered, such as cost, volume, weight, etc. Therefore, in this paper, we adopt multiple thresholds to mine erasable itemsets. Different thresholds are given to individual items to meet the requirements of practical applications. Besides, we adopt the loose constraint to decide the threshold value of each itemset with more than one item, such that more erasable itemsets can be derived as candidates to decision makers.

The rest of this paper is organized as follows. Section II describes the related work. Section III gives the problem definition. Section IV explains the sorted closure property. Section V goes into the finer steps of the proposed algorithm. Section VI describes the experiments, and Section VII gives the conclusion.

II. RELATED WORK

Erasable-itemset mining aims to identify combinations of materials that are not procured, allowing a factory to control loss within an acceptable range. This problem was introduced by Deng et al. in 2009, who also proposed a method called the META (Mining Erasable iTemsets with the Anti-monotone property) algorithm [8]. META employs an searching approach similar to the Apriori algorithm [1][2], progressively discovering all erasable itemsets layer by layer. It uses the downward-closure property effectively to reduce the search space of candidate itemsets. It, however, requires multiple database scans, leading to a lot of execution time. Subsequent to META, several improved algorithms have been introduced to address this issue, including VME (Vertical-format-based algorithm for Mining Erasable itemsets) [7], MERIT (fast Mining ERasable ITemsets) [9], MERIT+ (MERIT enhanced) [23], dMERIT+ (using Difference of NC_Set to enhance MERIT) [23], MEI (Mining Erasable Itemsets) [22], and BREM (Bitmap-Representation Erasable Mining) [17].

Copyright (c) IARIA, 2024. ISBN: 978-1-68558-144-2

VME was proposed by Deng et al. in 2010 [7]. It employed a list structure, storing additional information within PID_list to reduce the database scans to only twice, resulting in faster execution time than META. However, each itemset had its own dedicated PID_list, leading to substantial memory usage. Hong et al. proposed an enhancement to VME called BERM [17]. BERM utilized bit vectors to simplify the recording of the PID_list.

In 2012, Deng et al. presented a tree-based algorithm called MERIT [9]. It initially constructed a WPPC (Weighted Pre-Post Code) tree in an FP-growth [10] manner and then calculated NC-sets for each itemset using the WPPC tree. The NC-sets were leveraged to reduce memory usage and enhance execution speed.

MERIT+ was subsequently introduced by Le et al. [23], building upon the original MERIT and incorporating a weighted index to solve the issue of not mining all erasable itemsets. However, the duplicate NC-sets increased memory usage. Then, dMERIT+ [23] adopted a new structure called dNC_sets and used a hash table to eliminate redundant information, optimizing both memory and execution time. Le et al. then proposed the MEI approach, which employed a depth-first search strategy and the dPID_set structure.

In recent years, numerous derivative problems and applications related to erasable mining have been continuously proposed [24][28]. A factory engages in the production of a diverse range of products, considering various additional factors. For example, each product may require a different quantity of materials [15], and some products may experience peak sales only within specific time frames [11][12][13][20]. The ordering sequence from customers is another consideration for certain products [16]. Over time, the variety of products produced by a factory may increase [6][21][25][27], or a factory may face product discontinuation [18][19]. Incremental erasable-itemset mining avoids the need to re-run erasable mining every time a new product is added or removed. It becomes crucial to perform mining selectively on only the itemsets that have an impact rather than the entire database. Different mining problems incessantly appear for actual applications.

III. PROBLEM DESCRIPTION

Erasable-itemset mining is utilized in the management of manufacturing factories, where various products are produced, as illustrated in Table 1.

TABLE 1: AN EXAMPLE OF A PRODUCT DATABASE.

Product Database		
PID	*Items*	*Profit*
Product 1	*ABE*	200
Product 2	*DEF*	200
Product 3	*BCE*	100
Product 4	*ADF*	100
Product 5	*BF*	300
Product 6	*ACDF*	100

In the product database, each product consists of three fields: PID, Items, and Profit. The PID (product identification) serves as a code to distinguish different products. The items represent the materials required to produce a product, and the profit is the earnings obtained after selling the product.

Let us envision a scenario where a factory encounters challenging situations, such as a decline in financial resources, limited logistics transportation capacity, or insufficient storage space for materials. This leads to the inability to procure production from all the raw materials, necessitating a decision on which materials to erase. Consequently, products reliant on these erased materials cannot be manufactured, causing the factory to be unable to sell them and resulting in a decline in profits. The challenge is to determine which materials not to purchase, thereby controlling losses within acceptable proportions for the factory. This problem is known as erasable mining, and the different combinations of materials identified in this process are termed erasable itemsets. Subsequently, we will provide detailed definitions for the relevant terms associated with erasable mining.

Definition 1: (Multiple maximum thresholds) The user or factory presets a value between 0 and 1 according to the characteristics of each item, which is used to represent the percentage of the maximum loss in the total revenue that the user or factory can accept if the item is not restocked. The maximum threshold value of each item is expressed as λ. An example is given in Table 2, in which $\lambda(A) = 0.6$ and $\lambda(B) = 0.5$.

TABLE 2: THE MAXIMUM THRESHOLDS OF THE ITEMS IN THE ABOVE EXAMPLE.

Item	*A*	*B*	*C*	*D*	*E*	*F*
λ	0.6	0.5	0.4	0.7	0.3	0.8

Definition 2: (Maximum gain threshold) The maximum gain threshold represents the maximum practical loss acceptable to a user or a factory. The maximum gain threshold of an item *X* is expressed as *MGT(X)*, defined as follows:

$$MGT(X) = Profit_{Total} \times \lambda(X).$$

Take the items *A* and *B* as examples. The total profit of the whole set of products is 200 + 200 + 100 + 100 + 300 + 100 = 1000. According to the maximum threshold of each item in Table 2, *MGT(A)* = 1000 * 0.6 = 600, and *MGT(B)* = 1000 * 0.5 = 500.

Definition 3: (Gain) When a particular material cannot be purchased or stocked, the products that need to be produced with this material will be unable to be manufactured. The losses caused by these products that cannot be manufactured are called gains. The gain of the itemset *X* is expressed as *gain(X)* defined as follows:

$$gain(X) = \sum_{\{P | X \cap P.Items \neq \emptyset, P \in PD\}} P.Profit$$

where *P* is a product in the product database *PD*.

In multiple-threshold mining, the maximum thresholds of items are not the same. Different itemsets have their own distinct maximum thresholds. When an itemset contains only one item, we use its given threshold. However, when an itemset consists of two or more items, the calculation of the maximum threshold depends on different constraints. In this paper, we adopt the loose constraint for determining these thresholds. The formula for the loose constraint is defined as follows:

$$\lambda(X) = max(\lambda(I) | \ I \in X).$$

For example, the 2-itemset {*D, E*} contains both items *D* and *E*. Its maximum threshold is then set as *max(λ(D), λ(E)*), which is *max*(0.5, 0.4) = 0.5. Simultaneously, this constraint can also be applied to determine the maximum gain threshold.

Downward closure is a useful property in data mining and has been successfully applied in various mining algorithms, including the tight constraint in multiple threshold mining. However, this property does not apply to the loose constraint. A simple example illustrates this. In Table 1, *Gain(DE)* = 700 ≤ *MGT(DE)* = 700. {*D, E*} is thus an erasable itemset. But the gain of its subset *Gain(E)* = 400 > *MGT(E)* = 300. Thus, {*E*} is not an erasable itemset. Hence, the loose constraint does not possess the downward-closure property.

IV. Sorted-Closure Property

As mentioned above, the multiple-threshold mining with the loose constraint does not have the downward-closure property. Liu et al. proposed a novel technique called the sorted-closure property to replace the downward-closure property and successfully applied it in frequent itemset mining with multiple minimum supports [26]. In the traditional erasable-itemset mining algorithm, each item in an itemset is usually arranged according to the linguistic order or numerical order, such as {*A, B, C*} or {A_1, A_2, A_3}. However, for using the sorted-closure property, items will be sorted in the descending order of their thresholds. Through such changes, some useful properties will be produced, and unpromising candidate itemsets can be effectively pruned off. Thus, the time spent on scanning databases for calculating gains and verification can be reduced.

For example, the items in Table 2 are sorted in descending order according to their maximum thresholds. The sorted result is shown in Table 3. According to this order, the itemset {*D, E, F*} is rearranged as {*F, D, E*}.

TABLE 3: THE SORTED ITEMS ACCORDING TO THEIR MAXIMUM THRESHOLDS.

Item	*F*	*D*	*A*	*B*	*C*	*E*
λ	0.8	0.7	0.6	0.5	0.4	0.3

The following four theorems can then be deduced for the sorted items. By employing these theorems, we can effectively reduce the search space of candidate itemsets.

Theorem 1 Assume item *X* is an erasable 1-itemset and an item *Y*, after sorting, is located before item *X*. The gain of the 1-itemset *X* must be less than or equal to the maximum gain threshold of the 1-itemset *Y*. That is, *gain(X)* ≤ *MGT(Y)*.

Theorem 2 Assume itemset *X* is an erasable 2-itemset under the loose constraint and represented as {$item_1$, $item_2$}, where $item_1$ is located before $item_2$ after sorting. Then the following two conditions must be satisfied:

$gain(item_1) \le MGT(item_1)$, and
$gain(item_2) \le MGT(item_1)$.

Theorem 3 Assume itemset *X* is an erasable *k*-itemset (*k* ≥ 3) under the loose constraint with the first and the second items in *X* having different maximum gain thresholds. Then, the (*k*-1)-subitemsets of *X* containing the first item in *X* must also be erasable under the loose constraint.

Theorem 4 Assume itemset *X* is an erasable *k*-itemset (*k* ≥ 3) under the loose constraint with the first and the second items in *X* having the same maximum gain thresholds. Then, all the (*k*-1)-subitemsets of *X* must also be erasable under the loose constraint.

V. The Proposed Algorithm under the Loose Constraint

In this section, we will introduce the proposed algorithm for finding erasable itemsets under the loose constraint. It uses the four theorems mentioned above to reduce candidate itemsets. The algorithm is described below.

STEP 1: Sort the items according to the descending order of their maximum thresholds.

STEP 2: Scan the whole database to calculate the total gain of the database and the gain of each item.

STEP 3: Calculate the maximum gain threshold of each item using the total gain of the database.

STEP 4: Check the sorted items one by one from the front until the first one *X* with its gain smaller than or equal to its maximum gain threshold; Put *X* in the set of candidate 1-itemsets CI_1.

STEP 5: Check the sorted items after the above *X* one by one; If the gain of an item is smaller than or equal to the maximum gain threshold of *X*, add the item to CI_1.

STEP 6: Check whether the gain of each candidate 1-itemset in CI_1 is less than or equal to its own maximum gain threshold. If the 1-itemset conforms to the above condition, put it into the set of erasable 1-itemsets EI_1.

STEP 7: If EI_1 is empty, then no erasable itemsets are found, and the algorithm stops.

STEP 8: Form candidate 2-itemsets CI_2 by joining the candidate 1-itemsets in CI_1.

STEP 9: For each candidate 2-itemset in CI_2 with its first item being an erasable 1-itemset in EI_1, check whether its gain is less than or equal to its maximum gain threshold. If the itemset conforms to the above condition, put it into the set of erasable 2-itemsets EI_2.

STEP 10: If EI_2 is empty, then only erasable 1-itemsets are found; we output EI_1 and stop the algorithm.

STEP 11: Set k =3, where k is used to represent the number of items in an itemset to be processed.

STEP 12: Find any two itemsets in EI_{k-1} with the same (k-2) items to join and generate a k-itemset X. If the first two items (according to the sorted list) in X have different maximum gain thresholds, then add X to the candidate k-itemsets CI_k only if all the (k-1)-subitemsets of X containing the first item of X are in EI_{k-1}; If the first two items in X have the same maximum gain thresholds, then add X to CI_k only if all the (k-1)-subitemsets of X are in EI_{k-1}.

STEP 13: Check whether the gain of each candidate k-itemset in CI_k is less than or equal to its maximum gain threshold. If the itemset conforms to the above condition, put it into the set of erasable k-itemsets EI_k.

STEP 14: If EI_k is empty, then erasable 1-itemsets to (k-1)-itemsets are the desired; we output them and stop the algorithm; Otherwise set k as k+1 and go to STEP 12.

Note that in STEP 12, CI_k is formed effectively according to Theorems 3 and 4.

VI. Experimental Results

To enhance clarity in assessing the algorithm's performance, we conducted a comparison between multiple thresholds under the loose and the tight constraints [14] and under single thresholds [8] in our experiments. The program was executed on a system with a CPU i7-9750H@2.60GHz, 8GB RAM, and Windows 10. The programming language used was Java 19.0.1. The synthetic dataset P100KI0.05KD10 was generated by the IBM data generator [3] and named based on its parameters, where P represents the number of products, I represents the number of items, and D represents the average number of materials in each product. However, the data generated by the IBM data generator lacked profit information. To better simulate real-world scenarios, we thus introduced profit values using a normal distribution, denoted as $N(100, 20)$, where the two parameters represent the mean and the standard deviation, to ensure most product profits fall within a moderate range. Excessively high profits could result in unmarketable high prices, while excessively low profits might affect factory operations.

Concerning the thresholds set in the experiments, we initially defined an interval, generating the thresholds of the items uniformly within the range, denoted as $U(H, L)$. Here, H is the highest threshold, and L is the lowest within the interval. H and L are then used as the two thresholds, which were set, respectively, in the single-threshold algorithm. In the experiments, we first maintained other parameters constant while incrementally increasing each threshold by 0.01. The variation allowed us to observe its influence on the algorithm.

Figures 1 and 2 represent the quantities of erasable itemsets and candidate itemsets, respectively. Erasable itemsets are derived from the candidate itemsets. Hence, the number of erasable itemsets is always less than that of candidate itemsets. As the threshold values increase, both the quantities exhibit an upward trend. The two figures show that the quantities of the erasable itemsets and the candidate erasable itemsets under the tight and the loose constraints in multi-threshold mining fall between those of the single-threshold algorithm under two single thresholds. Specifically, the loose constraint tends to be closer to the highest single threshold, while the tight constraint tends to be closer to the lowest single threshold.

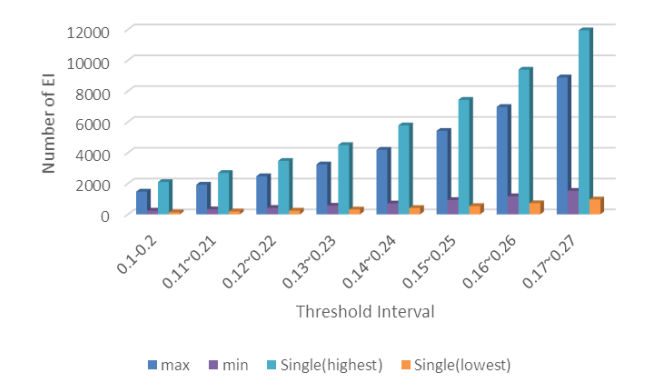

Figure 1. The numbers of erasable itemsets for different threshold intervals.

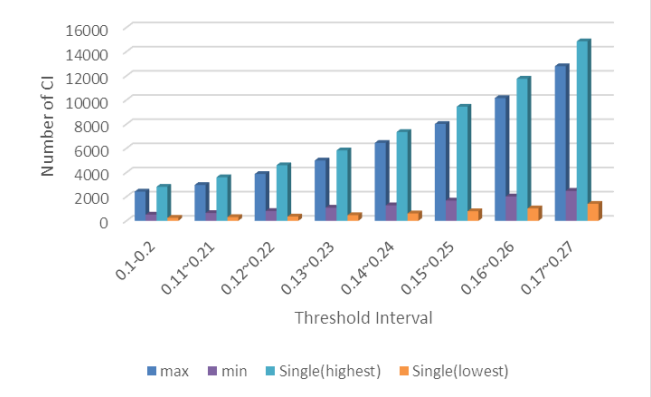

Figure 2. The numbers of candidate itemsets for different threshold intervals.

Figure 3 illustrates the memory usage for each program. The extent of memory usage is determined by the number of products, erasable itemsets, and candidate itemsets. As the number of products remains constant, the primary influencing factors are the quantities depicted in Figure 1 (erasable itemsets) and Figure 2 (candidate itemsets). With an increase in the quantities of both the erasable and candidate itemsets, the memory usage also increases accordingly.

Copyright (c) IARIA, 2024. ISBN: 978-1-68558-144-2

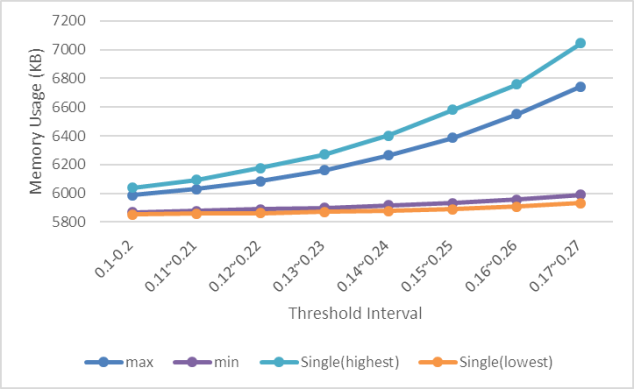

Figure 3. The memory usage for different threshold intervals.

Lastly, Figure 4 represents the execution time for each program. The most time-consuming aspect during the mining process is the generation and validation of candidate itemsets, which are closely related to the number of candidate itemsets, as shown in Figure 2. As the number of candidate itemsets increases, the execution time also rises accordingly.

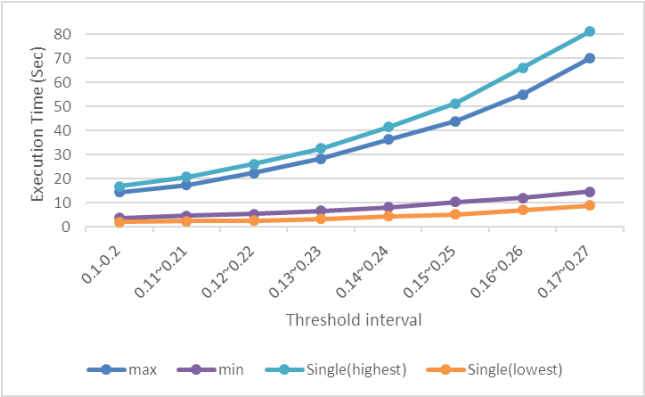

Figure 4. The execution time for different threshold intervals.

VII. Conclusion and Future Work

In this paper, we have proposed an algorithm for the loose constraint in multiple-threshold mining. We have also demonstrated the applicability of relevant theorems derived from the sorted closure, successfully solving the issue of lacking downward closure in the loose constraint. In our experiments, we have not only compared the algorithms with different constraints but also contrasted them with the single-threshold META algorithm. We have also comprehensively analyzed execution time, memory usage, erasable-itemset quantity, and candidate-itemset quantity. We will continuously explore designing algorithms tailored to various constraints for future research. Optimizing the execution time of multi-threshold mining could also be a focus of further investigation.

Acknowledgment

This work was supported by the National Science and Technology Council, Taiwan, under the grant NSTC 112-2221-E-390-014-MY3.

References

[1] R. Agrawal, T. Imieliński, 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.

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

[3] R. Agrawal and R. Srikant, *Quest Synthetic Data Generation Code*. 1994.

[4] R. Agrawal and R. Srikant, “Mining sequential patterns,” *The Eleventh International Conference on Data Engineering*, pp. 3-14, 1995.

[5] R. Chan, Q. Yang, and Y.-D. Shen, “Mining high utility itemsets,” *The Third IEEE International Conference on Data Mining*, pp. 19-19, 2003.

[6] R. Davashi, “IME: efficient list-based method for incremental mining of maximal erasable patterns,” *Pattern Recognition*, vol. 148, pp. 110166, 2024.

[7] Z. Deng and X. Xu, “An efficient algorithm for mining erasable itemsets,” *The International Conference on Advanced Data Mining and Applications*, pp. 214-225, 2010.

[8] Z. H. Deng, G. D. Fang, Z. H. Wang, and X. R. Xu, “Mining erasable itemsets,” *The 2009 International Conference on Machine Learning and Cybernetics*, pp. 67-73, 2009.

[9] Z. H. Deng and X. R. Xu, “Fast mining erasable itemsets using NC_sets,” *Expert Systems with Applications*, vol. 39(4), pp. 4453-4463, 2012.

[10] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” *ACM Sigmod Record*, vol. 29(2), pp. 1-12, 2000.

[11] T.-P. Hong, J.-X. Li, Y.-C. Tsai, and W.-M. Huang, “Tree-based unified temporal erasable-itemset mining,” *The Asian Conference on Intelligent Information and Database Systems*, pp. 224-233, 2023.

[12] T. P. Hong, H. Chang, S. M. Li, and Y. C. Tsai, “A unified temporal erasable itemset mining approach,” *The 2021 International Conference on Technologies and Applications of Artificial Intelligence*, pp. 194-198, 2021.

[13] T. P. Hong, H. Chang, S. M. Li, and Y. C. Tsai, “A dedicated temporal erasable-itemset mining algorithm,” *The International Conference on Intelligent Systems Design and Applications*, pp. 977-985, 2022.

[14] T. P. Hong, Y. C. Chang, W. M. Huang, and W. Y. Lin, “Multiple-threshold erasable mining under the tightest constraint,” *The International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 369-377, 2022.

[15] T. P. Hong, H. W. Chen, W. M. Huang, and C. H. Chen, “Erasable pattern mining with quantitative information,” *The 2019 International Conference on Technologies and Applications of Artificial Intelligence*, pp. 1-4, 2019.

[16] T. P. Hong, Y. L. Chen, W. M. Huang, and Y. C. Tsai, “Erasable-itemset mining for sequential product databases,” *International Conference on Hybrid Intelligent Systems*, pp. 566-574, 2022.

[17] T. P. Hong, W. M. Huang, G. C. Lan, M. C. Chiang, and J. C. W. Lin, “A bitmap approach for mining erasable itemsets,” *IEEE Access*, vol. 9, pp. 106029-106038, 2021.

[18] T. P. Hong, C. C. Li, S. L. Wang, and C. W. Lin, “Maintenance of erasable itemsets for product deletion,” *The Fifth Multidisciplinary International Social Networks Conference*, pp. 1-4, 2018.

[19] T. P. Hong, C. C. Li, S. L. Wang, and J. C. W. Lin, “Reducing database scan in maintaining erasable itemsets from product

Copyright (c) IARIA, 2024. ISBN: 978-1-68558-144-2

deletion," *The 2018 IEEE International Conference on Big Data*, pp. 2627-2632, 2018.

[20] T. P. Hong, J. X. Li, Y. C. Tsai, and W. M. Huang, "Unified temporal erasable itemset mining with a lower-bound strategy," *The 2022 IEEE International Conference on Big Data*, pp. 6207-6211, 2022.

[21] T. P. Hong, K. Y. Lin, C. W. Lin, and B. Vo, "An incremental mining algorithm for erasable itemsets," *The 2017 IEEE International Conference on Innovations in Intelligent Systems and Applications*, pp. 286-289, 2017.

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

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

[24] T. Le, B. Vo, and G. Nguyen, "A survey of erasable itemset mining algorithms," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 4(5), pp. 356-379, 2014.

[25] G. Lee, U. Yun, H. Ryang, and D. Kim, "Erasable itemset mining over incremental databases with weight conditions," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 213-234, 2016.

[26] B. Liu, W. Hsu, and Y. Ma, "Mining association rules with multiple minimum supports," *The Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 337-341, 1999.

[27] H. Nam, U. Yun, E. Yoon, and J. C. W. Lin, "Efficient approach for incremental weighted erasable pattern mining with list structure," *Expert Systems with Applications*, Vol. 143, pp. 113087, 2020.

[28] D. M. D. Raj and M. Ranganathan, "A comprehensive survey on erasable itemset mining," *International Journal of Computer Science and Information Security*, vol. 15(7), pp. 184-201, 2017.

Copyright (c) IARIA, 2024. ISBN: 978-1-68558-144-2