Single Phase High Utility Pattern Mining for Dynamic Datasets

Abhinav Agnihotry[1] Gurpreet Singh[2] Pallavi Purohit[3] Snehita P. Chhabria[4]
[1,2,3,4]B.E. Student
[1,2,3,4]Department of Computer Science & Engineering
[1,2,3,4]VTU Belgavi, India

Abstract— High utility mining is a new development of data mining technology. Previous solutions of this problem utilize a two-phase process with candidate generation approach that is inefficient and not scalable with large dynamic databases. Two-phase method suffers from scalability issue due to the huge number of candidates. This paper proposes a novel algorithm that finds high utility patterns in a single phase without generating candidates. The oddity lies in a high utility pattern growth approach, a lookahead strategy, and a linear data structure. Our pattern growth approach searches a reverse set enumeration tree and prune search space by utility upper bounding. It uses closure property and a singleton property to look ahead and identify high utility patterns without enumeration. A linear data structure enables us to compute a tight bound for powerful pruning and to directly identify high utility patterns in an efficient and scalable way, which targets the root cause with prior algorithms. Our algorithm can be 1 to 2 orders of magnitude more efficient and is more scalable than the traditional state-of-the-art algorithms.

Key words: Data Mining, Utility Mining, Frequent Itemsets, High Utility Itemsets

I. INTRODUCTION

Searching for unique patterns has a large number of applications, and has been an important task, some examples of the applications are, genome analysis, condition monitoring, cross marketing, and inventory prediction, where interestingness [7], [17], [21] and uniqueness measures play an important role. With frequent pattern mining, a pattern is considered as unique if its occurrence frequency exceeds a user-specified threshold which is also known as the Minimum Utility Threshold. Consider, mining frequent patterns from a shopping transaction database, it involves finding sets of products that are frequently purchased together by customers. However, we may also want to use a lot of other factors that are not given by the occurrence frequency. For example, a supermarket manager might want to list the products on the basis of their profits or income, which is in no way related to the occurrence frequency, therefore they are listed on the basis of quantities they are purchased in and the profits and these factors are not considered in frequent pattern mining.

Utility mining [21] came up recently to overcome the disadvantages of frequent pattern mining by considering the user's expectation or aim as well as the raw input data. Utility mining with the itemset share framework [9], [19], [20], for example, identifying sequence of products with high revenues or profits, is much harder to remove than other categories of utility mining problems. Some of the other disadvantages are, objective-oriented utility-based association mining [5], [16] and weighted itemset mining [4], [12], [14]. Concretely, the uniqueness measures in the prior categories exhibit an anti-monotonous property. Such a property can be used to prune or reduce search spaces, which is also the base of all frequent pattern mining algorithms [2]. Unfortunately, the anti-monotonous property is not applicable to utility mining with the itemset share framework [19], [20]. Therefore, utility mining with the itemset share framework is more difficult than the other categories of utility mining including frequent pattern mining.

Most of the existing utility mining algorithms with the item-set share framework [3], [6], [11], [13], [18], [19] make use of a two-phase, candidate generation approach, i.e, in the first phase the algorithm finds candidates of high utility patterns, and then in the second phase it scans the raw data one more time to identify high utility patterns from the candidates that were generated in the first phase.

The main limitation is that the candidates can be huge in number, which is known as the scalability and efficiency bottleneck. Even though a lot of effort has gone into [3], [6], [11], [18] reducing the number of candidates generated in the first phase, the problem still persists when the raw input data may contain many long transactions or the minimum utility threshold value is less. These large number of candidates causes scalability issue not only in the first phase but also in the second phase, and continuously degrades the efficiency. This is overcome in the HUI Miner algorithm, however it is way less efficient than two-phase algorithms when it comes to mining huge databases due to inefficient join operations, lack of strong pruning and reduction of search spaces, and scalability issue with its vertical data structure. To overcome this difficulty, this paper proposes the modified version of the algorithm, d2HUP, for utility mining with the itemset share framework, which uses several techniques that can be used for mining frequent patterns, including searching a regular set enumeration in a reverse lexicographic order [22] and heuristics for arranging and ordering items [8], [22].

II. RELATED WORKS

Frequent-itemset mining has been studied extensively in the literature and several algorithms have been proposed. High-utility itemset mining finds itemsets from the database which have their utility no less than a minimum utility threshold. Several algorithms have been proposed to find high utility itemsets.

A. High Utility Pattern (HUP) mining

C. F. Ahmed, S. K. Tanbeer [3] discussed high utility pattern (HUP) mining as being one of the most important research issues in data mining due to its ability to consider the non-binary frequency values of items in transactions and different profit values for all items. This paper proposed three novel tree structures to perform incremental and interactive HUP mining.

B. ExAnte: Approach for preprocessing input data

F. Bonchi [6] discussed that ExAnte is a simple yet effective approach for preprocessing input data for mining frequent patterns. The approach questions established research in that it requires no trade-off between ant monotonicity and monotonicity. Together, they brought about an extremely proficient successive itemset mining calculation that viably abuses monotone constraints.

C. High Transaction-Weighted Utilization Itemsets

S. Krishnamoorthy [10] depicted the utility mining process where the possible candidate itemsets are generated from high transaction-weighted utilization itemsets of the preceding level. The items not present in the set of candidates itemsets at the current level are not the members of high transaction-weighted utilization itemsets in later levels. These items are thus removed from the transactions. Once transactions are modified in each level, the items in some transactions may be the similar. The transactions with the same items can then be merged into one. The quantity of each item in the final transaction is the summation of all its quantities.

D. Single Phase without Generating Candidates

Benjamin C.M., Liu, Ke Wang [1] proposes a novel algorithm that finds high utility patterns in a single phase without generating candidates. The novelties lie in a high utility pattern growth, a lookahead strategy, and a linear data structure. Pattern growth approach is to search a reverse set tree and to prune search space by utility upper bounding.

III. METHODOLOGY

This section proposes a new approach to the problem, that is, a high utility pattern growth approach. Firstly introduce a reverse set enumeration tree as a way to enumerate patterns, and then propose strong pruning techniques that drastically reduces the number of patterns to be enumerated, which lays the theoretical foundation for the proposed algorithm.

A. Pattern Growth

The natural path to mine high utility pattern is to enumerate each subset pattern X in universe of items I, and check if X has a utility over the threshold given. Strong pruning techniques are employed as an intensive enumeration is unwise due to the huge number of subsets of I.

The section introduces an approach to the problem, that is, a high utility pattern growth approach. A reverse set enumeration tree is introduced as a way to enumerate patterns, and then strong pruning techniques that reduces the number of patterns to be enumerated, which lays the base for our algorithm.

B. Reverse Set Enumeration Tree

Direct Discovery of High Utility Patterns, which is an integration of the depth-first search of the reverse set enumeration tree, which prune the techniques that drastically reduces the number of patterns to be enumerated, and a novel data structure that enables efficient computation of utilities and upper bounds. We are using depth-first strategy because breadth-first search is typically more memory intensive and more likely to exhaust main memory and thus it is slower.

The reverse set enumeration tree is equivalent to a regular set enumeration tree [1], [8], [15] that is imposed with a reverse lexicographic order [22] and explored right-to-left, which yields a property: a pattern is always enumerated before its supersets [22] in a depth-first search.

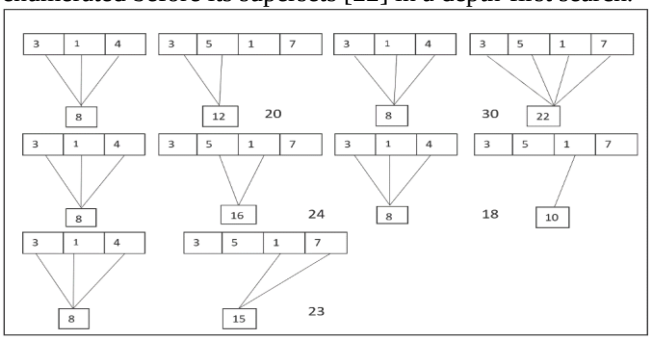

Fig. 1: Representation of Prefix and suffix in the value-pattern format

C. Utility Upper Bounding

In the past, many algorithms have been designed to mine high average-utility itemsets the upper-bound model was proposed to prune unpromising itemsets, which was used in many studies to improve the performance. The estimated upper-bound utility of an itemset X is defined as the sum of the transaction-maximum utilities (tmu) of transactions containing X in the database. In accordance with pattern growth approach, it is to estimate an upper bound on utilities of all possible patterns explored, when growing the reverse set enumeration tree. If these do not satisfy the minimum utility, the parent node along with its child nodes can be predicted as not high utility patterns.

D. Look ahead to avoid Enumeration

Looking ahead in a search process is valuable even if it incurs little extra computation cost. Similar to closed frequent pattern mining [23], we can observe that when all the prefixes have the same support, in particular for two cases, therefore it is inexpensive to look ahead using the following cases.

1) Case 1: Every prefix extension of a pattern has the same support and has a utility more than or equal to minU, which can be tested.
2) Case 2: All the prefix extensions of a pattern have the same support, but among which only the longest has a utility no less than minU. Such a case can be identified by Theorem 5 with little computation.

E. Proposed Algorithm

Algorithm: d^2 HUP Modified Algorithm
Input: D: Database XUT: External Utility Table minU: Threshold
Output: all high utility itemsets
build TS ({}) and Ω from D and XUT
N ← root of reverse set enumeration tree
DFS (N, TS(pat(N)), minU, Ω) Subroutine: DFS (N, TS(pat(N)), minU, Ω)
if u(pat(N)) ≥ minU then output pat(N)
W←{i|i < pat(N) uB_{item} (i, pat(N) ≥ minU)
if Closure(pat(N), W, minU) is satisfied
then output nonempty subsets of W ∪ pat(N)
else if Singleton(pat(N), W, minU) is satisfied
then output W ∪ pat(N) as an HUP

else foreach item i ϵ W in Ω do
if $uB_{fpe}(\{i\} \cup pat(N)) \geq minU$
then C← the child node of N for i
TS(pat(N)) ←Project(TS(pat(N)), i)
DFS (C, TS(pat(N)), minU, Ω)
end foreach
A ←New items in database
D2HUP(A, XUT, minU)

IV. RESULTS

In this section we will discuss about the result which we have got after implementing this algorithm. Our D2HUP algorithm has been executed with sample market transaction dataset and its utility table as an input. D2HUP algorithm scans the whole database in one phase only and find the itemsets which contains high utility patterns with respect to minimum utility which is considered as a threshold utility.

Algorithms	Input File Size (Byte)	Memory Consumption
H-Mine	10K	300 KB
THUI	10K	250 KB
Combined	10k	13 MB
D2HUP	100k	20 MB

Table 1: Execution time of various Algorithm

Algorithms	Input File Size (Byte)	Execution time
WARM	1000K	1000 Sec
Two-Phase	1000K	750 Sec
Apriori Hybrid	100K	7.5 Sec
Combined	100k	1500 Sec
D2HUP	100k	1.3 Sec

Table 2: Memory Consumption of various Algorithms

We evaluate our d2HUP algorithm by comparing with the state-of-the-art algorithms like TwoPhase [13], WARM, Apriori Hybrid and Combined Algorithm. The code of TwoPhase [13] and Apriori Hybrid were provided by the original authors. In Table, the outcome of various algorithms has been analysed with respect to memory consumption of respective algorithm. The table demonstrates input file size and relating execution time of various algorithms.

V. CONCLUSIONS

The paper proposes a modified form of the d2HUP algorithm, for utility mining with the itemset share framework, which finds high utility patterns without candidate generation in a single phase. Our contributions include 1) An efficient approach to process the data dynamically included into the database during runtime. 2) Our approach is enhanced significantly by the lookahead strategy that identifies high utility patterns without enumeration. In the future, we would like to work on high utility sequential pattern mining, parallel and distributed algorithms, and their application in big data analytics.

REFERENCES

[1] R. Agarwal, C. Aggarwal, and V. Prasad, "Depth first generation of long patterns," in Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2000, pp. 108–118.
[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in Proc. 20th Int. Conf. Very Large Databases, 1994, pp. 487–499.
[3] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "Efficient tree structures for high utility pattern mining in incremental databases,"IEEE Trans. Knowl. Data Eng., vol. 21, no. 12, pp. 1708–1721, Dec. 2009.
[4] C. H. Cai, A. W. C. Fu, C. H. Cheng, and W. W. Kwong, "Mining association rules with weighted items," in Proc. Int. Database Eng. Appl. Symp., 1998, pp. 68–77.
[5] R. Chan, Q. Yang, and Y. Shen, "Mining high utility itemsets," in Proc. Int. Conf. Data Mining, 2003, pp. 19–26.
[6] A. Erwin, R. P. Gopalan, "Efficient mining of high utility itemsets from large dataset," in Process. 12th Pacific-Asia Conf. Advance. Knowl. Discovery Data Mining, 2008, pp. 554–561.
[7] L. Geng and H. J. Hamilton, "Interestingness measures for data mining: A survey," ACM Comput. Surveys, vol. 38, no. 3, p. 9, 2006.
[8] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2000, pp. 1–12.
[9] R. J. Hilderman, C. L. Carter, H. J. Hamilton, and N. Cercone, "Mining market basket data using share measures and characterized itemsets," in Proc. PAKDD, 1998, pp. 72–86.
[10] S. Krishnamurthy, "Pruning strategy for mining high utility itemsets," Expert System. Application., vol. 42, no. 5, pp. 2371–2381, 2015.
[11] Y.-C. Li, J.-S. Yeh, and C.-C. Chang, "Isolated items discarding strategy for discovering high utility itemsets," Data Knowl. Eng., vol. 64, no. 1, pp. 198–217, 2008.
[12] T. Y. Lin, Y. Y. Yao, and E. Louie, "Value added association rules,"in Proc. 6th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining, 2002, pp. 328–333.
[13] Y. Liu, W. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," in Proc. Utility-Based Data Mining Workshop SIGKDD, 2005, pp. 253–262.
[14] S. Lu, H. Hu, and F. Li, "Mining weighted association rules," Intell. Data Anal., vol. 5, no. 3, pp. 211–225, 2001.
[15] J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "PrefixSpan: Mining sequential patterns efficiently by prefixprojected pattern growth," in Proc. 17th Int. Conf. Data Eng., 2001,pp. 215–224.
[16] Y. Shen, Q. Yang, and Z. Zhang, "Objective-oriented utility-based association mining," in Proc. IEEE Int. Conf. Data Mining, 2002, pp. 426–433.
[17] A. Silberschatz and A. Tuzhilin, "On subjective measures of interestingness in knowledge discovery," in Proc. ACM 1st Int. Conf. Knowl. Discovery Data Mining, 1995, pp. 275–281.
[18] S. Tseng, B.-E. Shie, C.-W. Wu, and P. S. Yu, "Efficient algorithms for mining high utility itemsets from transactional databases," IEEE Trans. Knowl. Data Eng., vol. 25, no. 8, pp. 1772–1786, Aug. 2013.
[19] H. Yao and H. J. Hamilton, "Mining itemset utilities from transaction databases," Data Knowl. Eng., vol. 59, no. 3, pp. 603–626, 2006.

[20] H. Yao, H. J. Hamilton, and C. J. Butz, "A foundational approach to mining itemset utilities from databases," in Proc. SIAM Int. Conf. Data Mining, 2004, pp. 482–486.
[21] H. Yao, H. J. Hamilton, and L. Geng, "A unified framework for utility-based measures for mining itemsets," in Proc. ACM SIGKDD 2nd Workshop Utility-Based Data Mining, 2006, pp. 28–37.
[22] M. J. Zaki, "Scalable algorithms for association mining," IEEE Trans. Knowl. Data Eng., vol. 12, no. 3, pp. 372–390, May/Jun. 2000.
[23] M. J. Zaki and C. Hsiao, "Efficient algorithms for mining closed itemsets and their lattice structure," IEEE Trans. Knowl. Data Eng., vol. 17, no. 4, pp. 462–478, Apr. 2005.
[24] Junqiang Liu, Ke Wang and Benjamin C.M. Fung, "Mining High Utility Patterns in One Phase without Generating Candidates", IEEE Trans. Knowl. Data Eng., vol. 17, 28, NO. 5, MAY 2016

