Efficient Method for Design and Analysis of Mining High Utility Itemsets

Sampada L. Mashirkar
M.Tech Student Fourth Semester CSE,
Priyadarshini Bhagwati
College of Engineering, Nagpur,
Maharashtra, India
mashirkar786@gmail.com

Archana A. Nikose
Assistant Professor, CSE,
Priyadarshini Bhagwati
College of Engineering, Nagpur,
Maharashtra, India
archananikose@gmail.com

Abstract* — Mining high utility itemsets from value-based databases is a vital information mining assignment, which alludes to the disclosure of itemsets with high utilities (e.g. high benefits). Albeit a few studies have been done, current techniques may display excessively numerous high utility itemsets for a client, which corrupts the execution of the mining errand as far as execution and memory proficiency. To accomplish high productivity for the mining undertaking and give a compact mining result to clients, we propose a novel system in this paper for mining closed+ high utility itemsets, which serves as a reduced and lossless representation of high utility itemsets. We display a proficient calculation called CHUD (Closed+ High Utility itemset Disclosure) for mining closed+ high utility itemsets. Further, a strategy called DAHU (Derive All High Utility itemsets) is proposed to recuperate all high utility itemsets from the arrangement of closed+ high utility itemsets without getting to the first database. Aftereffects of tests on genuine and manufactured datasets demonstrate that CHUD and DAHU are exceptionally effective with a gigantic decrease (up to 800 times in our tests) in the number of high utility itemsets. Also, when all high utility itemsets are recuperated by DAHU, the methodology joining CHUD and DAHU additionally beats the best in class calculations in mining high utility itemsets.

***Keywords*— Frequent itemset, closed high utility itemset, lossless and concise representation, utility mining, data mining.**

I. INTRODUCTION

Incessant itemset mining (contracted as FIM) [1, 10] is a key exploration subject in information mining. One of its well-known applications is business sector wicker container examination, which alludes to the disclosure of sets of things (itemsets) that are oftentimes acquired together by clients. Be that as it may, in this application, the customary model of FIM may find a vast measure of continuous itemsets with low benefit and lose the data on significant itemsets having low offering frequencies. These issues are brought about by the realities that (1) FIM regards all things as having the same significance/unit benefit/weight and (2) it accept that each thing in an exchange shows up in a parallel structure, i.e., a thing can be either present or missing in an exchange, which doesn't demonstrate its buy amount in the exchange. Consequently, FIM can't fulfil the prerequisite of clients who yearning to find itemsets with high utilities, for example, high benefits.

To address these issues, utility mining [2, 5, 6, 7, 8, 11, 13, 15, 19, 20, 24, 26] develops as an essential point in information mining. In utility mining, everything has a weight (e.g. unit benefit) and can seem more than once in every exchange (e.g. buy amount). The utility of an itemset speaks to its significance, which can be measured as far as weight, benefit, cost, amount or other data relying upon the client inclination. An itemset is known as a high utility itemset (abridged as HUI) if its utility is no not exactly a user specified least utility limit. Utility mining has an extensive variety of utilizations, for example, site click stream investigation [2, 5, 19, 24], cross-showcasing examination [6, 20, 26] furthermore, biomedical areas [7].

In any case, HUIs mining is not a simple errand since the descending conclusion property [1, 10] in FIM does not hold in utility mining. The inquiry space can't be specifically pruned to discover HUIs as in FIM since a superset of a low utility itemset can be a high utility itemset. Numerous studies [2, 13, 15, 20, and 21] were proposed for mining HUIs, yet they frequently show a huge number of high utility itemsets to clients such that appreciation of the outcomes gets to be troublesome. In the meantime, the calculations get to be wasteful regarding time and memory necessity. Specifically, the execution of the mining assignment diminishes extraordinarily under low least utility limits or thick databases.

To decrease the computational expense in FIM while exhibiting less and more vital examples to clients, numerous studies created succinct representations, for example, free sets [3], no derivable sets [4], maximal itemsets [9] and shut itemsets[14, 16-18 22, 27]. These representations effectively lessen the arrangement of itemsets found, however they were created for successive itemset mining rather than high utility itemset mining.

In this way, an imperative exploration inquiry "Is it conceivable to consider a smaller and lossless representation of high utility itemsets enlivened by these representations to address the previously stated issues in HUI mining?"

Noting this inquiry decidedly is difficult. Building up a compact and finish representation of HUIs represents a few difficulties:

1. Coordinating ideas of compact representations from FIM into HUI mining may deliver a lossy representation of all HUIs or a representation that is most certainly not significant to the clients.

2. The representation may not accomplish a noteworthy decrease in the quantity of extricated examples to legitimize utilizing the representation.

3. Calculations for extricating the representation may not be effective. They might be slower than the best calculations for mining all HUIs.

4. It might be difficult to build up an effective strategy for recuperating all HUIs from the representation.

In this paper, we address these difficulties by proposing a consolidated and important representation of HUIs named Closed+ High Utility Itemsets (Closed+ HUIs), which incorporates the idea of shut itemset into HUI mining. Our commitments are four-fold in correspondence to determining the four difficulties said beforehand:

1. The proposed representation is lossless by utilizing another structure named utility unit exhibit that permits recuperating all HUIs and their utilities proficiently.

2. The proposed representation is additionally conservative. Tests demonstrate that it diminishes the quantity of itemsets by a few requests of greatness, particularly for datasets containing long HUIs (up to 800 times).

3. We propose a productive calculation, named CHUD (Closed+ High Utility itemset Discovery), to discover this representation. It incorporates three novel systems named REG, RML and DCM that incredibly improve its execution. Results demonstrate that CHUD is much quicker than current best techniques for mining all HUIs [20].

4. We propose a top-down technique named DAHU (Derive All High Utility itemsets) for productively recuperating all HUIs from the arrangement of Closed+ HUIs. The blend of CHUD and DAHU gives another approach to get all HUIs and it outflanks UPGrowth [20], the state-of-the- craftsmanship calculation for mining HUIs.

The rest of this paper is sorted out as takes after. In Segment II, we present the foundation for smaller representations and utility mining. Area III characterizes the representation of closed+ HUIs and presents our techniques. Examinations are appeared in Section IV and conclusions are given in Section V.

TABLE I. AN EXAMPLE TRANSACTIONAL DATABASE

TID	Transaction	TU
T_1	A(1), B(1), E(1), W(1)	5
T_2	A(1), B(1), E(3)	8
T_3	A(1), B(1), F(2)	8
T_4	E(2), G(1)	5
T_5	A(1), B(1), F(3)	11

TABLE II. UNIT PROFITS FOR EVERY ITEM

Item	A	B	E	F	G	W
Unit Profit ($)	1	1	2	3	1	1

II. BACKGROUND

In this section, we review the existing methods for high utility itemset mining and generation of no redundant association rules in support-confidence framework. In the support-confidence framework, the non-redundant association rules are generated from the frequent closed itemsets and their generators. We also review existing works on generation of frequent closed itemsets together with their generators as it is essential for generation of non-redundant association rules.

A. *High utility itemset mining*

The conventional affiliation principle mining strategies [1] depend on bolster certainty system, where all things are considered with the same level of significance. The strategies proposed in to concentrate affiliation standard tail this traditional factual estimation delivering the same result on a given least backing and least certainty. The weighted affiliation principle mining (WARM) sums up the conventional structure by offering significance to things, where significance is given as weights.

Ramkumar et al. [36] presented the idea of weighted backing of itemsets and weighted affiliation rules on the premise of costs relegated to both things and exchanges. Later, considering just the thing Weights into record, Cai et al. [7] proposed the weighted backing of affiliation standards. Be that as it may, the weighted backing of the affiliation rules does not fulfill the descending conclusion property, which results in the execution corruption. With a specific end goal to overcome such issue, by considering exchange weight, Tao et al. [44] gave the idea of weighted descending conclusion property. Considering both backing what's more, weight of itemsets, Yun [55] then introduced another system, called the weighted fascinating example mining (WIP). Pears et al. [33] further proposed a WARM strategy that mechanizes the procedure of weight task to the things by figuring a straight model.

In WARM framework, note that the amounts of things in exchanges are not considered. Considering things' amounts in exchanges and their individual significance, high utility itemset mining (HUIM) gotten an impressive exploration consideration. Yao et al. proposed a scientific model of utility mining by summing up the offer certainty model[5]. As utility mining does not satisfy the descending conclusion property, Liu et al. proposed the two phase calculation that uses the exchange weighted descending conclusion property to prune the competitor high utility itemsets in the main stage and after that all the complete arrangements of high utility itemsets are acquired in the second stage. To decrease the quantity of hopeful itemsets in the primary stage, Li et al. moreover proposed a disconnected things disposing of methodology (IIDS) to the level-wise utility mining technique.

Ahmed et al. [3] proposed a FP-Growth based calculation that uses a tree structure, named IHUP Tree, furthermore, proficiently produces the hopeful high utility itemsets for incremental and the intelligent mining. To encourage decrease the quantity of itemsets in the principal stage, Tseng et al. proposed the tree-based techniques, named the UP-Growth and UP-Growth+, which utilize a few systems to diminish the evaluated utility estimation of an itemset, and thus, they upgrade the execution. To maintain a strategic distance from the level shrewd applicant era and test procedure, Song et al. proposed a simultaneous calculation, called the CHUI-Mine, for mining HUIs from exchange databases utilizing their proposed information structure CHUI-Tree to keep up the data of HUIs. Their proposed calculation produces the potential high utility itemsets utilizing two simultaneous procedures: the main procedure is utilized for development and element pruning the tree, and after that setting the contingent trees into a cradle, and the second one for perusing the restrictive design list from the cradle and mining HUIs. To accelerate the execution and diminish the memory prerequisite in the mining procedure, Lan et al. proposed a proficient utility mining approach that embraces a projection-based indexing system that specifically produces the required itemsets from the exchanges database. Ahmed et al. [2] proposed a novel tree-based competitor pruning system, called the High Utility Candidates Prune (HUC-Prune), for maintaining a strategic distance from more database sweeps and the level-wise candidate generation.

To maintain a strategic distance from the computational expense of competitor era and utility calculation, Liu and Qu then proposed an information structure, named the utility-rundown, to store both the utility data around an itemset and the heuristic data for pruning the inquiry space. Utilizing the built utility-records from a mined database, they built up an effective calculation, called the HUI-Miner, which mines high utility itemsets without applicant era in a profundity first pursuit way. Their calculation works in a solitary stage by specifically recognizing high utility itemsets in an effective way and it is likewise versatile. To decrease the expense of join operation in the computation of the utility-rundown of an itemset in HUI-Miner, Fournier-Viger et al. [14] enhanced the HUI-Miner consolidating with the things co-events technique (named as FHM) that is around six times speedier the HUI-Miner.

B. *Closed itemsets with their generators and non-redundant association rule mining*

To generate both frequent closed itemsets (FCI) and generators, Pasquier et al. [31] Proposed the CLOSE algorithm that is based on level-wise searching approach with the help of Apriori property. Szathmary et al. [41] proposed the ZART algorithm that generates FCIs with their generators in a level-wise manner. They further proposed the Eclat-Z algorithm [42] that mines frequent itemsets in a depth-first way and the FCIs with their generators are identifies in level-wise m anner. An effective method, named as Touch, was developed by combining the FCI method Charm [56] and the frequent generator (FG) mining algorithm, Talky-G [43]. The FCIs are mined using Charm and FGs are mined using Talky-G and, then Touch associates the generators to their closed itemsets using a suitable hash function.

Wu et al. [49] introduced the closer concept to high utility itemsets. They called the extracted itemsets as closed+ high utility itemsets. On incorporating closure based on support of itemsets, they proved, first mining the set of high utility itemsets and then applyin g closed constraint produces the same result while mining all the closed itemsets first and then applying t he utility constraint. They proposed an effective method named as CHUD (Closed+ High Utility itemset Discovery) for mining closed+ high utility itemsets. Further, they proposed a method called the DAHU (Derive All High Utility itemsets), to recover all high utility itemsets from the set of closed+ high utility itemsets without further accessing the database. In addition, they proposed AprioriHC and AprioriHC-D algorithms [50] and mentioned that CHUD performs better than AprioriHC and AprioriHC-D. However, no suitable method for high utility closed itemset with their generator is proposed for high utility itemset mining.

To reduce the number of association rules extracted in support confidence-framework, several meth-ods have been developed in the literature [4, 10, 21, 31, 32, 37, 41, 51, 52, 56]. Kryszkiewicz [21] proposed the representative association rules (RR) with the help of a cover operator that represents a set of association rules. Zaki [56] proposed a method to reduce the number of association rules and the extracted rules, called the general rules, which have shortest antecedent and shortest consequent giving an equivalence class of rules of same support and confidence. Pasquier et al. [32] Defined the min-max rules having minimal antecedent and maximal consequent. Their proposed method eliminates the non-redundant rules as min-max exact and min-max approximate rules from the frequent closed itemsets and their generators. Furthermore, to reduce more rules, Cheng et al. [10] Proposed the concept of δ-tolerance, which is a relaxation on the closure defined on the support of frequent itemset. Yahia et al. [52] Proposed an informative basis to reduce the number of association rules, which is further efficiently compressed by Sahoo et al. [37]. Xu et al. [51] Filtered the min -max rules by defining redundancy and provided the reliable exact basis and reliable approximate basis of the same inference capacity. Balc´azar [4] further obtained a small and crisp set of association rules by the help of confidence boost of a rule, which eliminates the rules with similar confidence.

III. IMPLEMENTATION

A. *The AprioriHC Algorithm*

Initially, a variable k is set to 1. The algorithm performs a database scan to compute the transaction utility of each transaction. At the same time, the TWU of each item is computed. Each item having a TWU no less than abs_min_utility is added to the set of 1-HTWUIs Ck. Then the algorithm proceeds recursively to generate itemsets having a length greater than k. During the kth iteration, the set of k-HTWUIs Lk is used to generate (k þ 1)-candidates Ckþ1 by using the Apriori-gen function [1]. Then the algorithm computes TWUs of itemsets in Ckþ1 by scanning the database Donce. Each itemset having a TWU no less than abs_min_utility is added to the set of (k þ 1)-HTWUIs Lkþ1. After that, the algorithm removes non-closed itemsets in Lkþ1 by the following process. For each candidate X in Lkþ1, the algorithm checks if there exists a subset Y ⍰ X such that Y 2 Lk and SCðXÞ ¼ SCðY Þ. If true, Y is deleted from Lk because Y is not a closedþ high utility itemset according to Definition 14. If false, Y is kept and marked as "closed" because it may be a closedþ high utility itemset. The phase I of AprioriHC terminates when no candidate is generated. Then, the algorithm performs Phase II. In phase II, the algorithm scans the database once and calculates the utilities of HTWUIs that are marked as "closed" to identify the set of closedþ high utility itemsets.

In data mining, Apriori is a classic algorithm for learning association rules. Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation).

The whole point of the algorithm (and data mining, in general) is to extract useful information from large amounts of data. For example, the information that a customer who purchases a keyboard also tends to buy a mouse at the same time is acquired from the association rule below:

Support: The percentage of task-relevant data transactions for which the pattern is true.

Support (Keyboard -> Mouse) =

$$\frac{\text{No. of transactions containing both Keyboard and Mouse}}{\text{No. of total transactions}}$$

Confidence: The measure of certainty or trustworthiness associated with each discovered pattern.

Confidence (Keyboard -> Mouse) =

$$\frac{\text{No. of transactions containing both Keyboard and Mouse}}{\text{No. of transactions containing (Keyboard)}}$$

The algorithm aims to find the rules which satisfy both a minimum support threshold and a minimum confidence threshold (Strong Rules).
Item: article in the basket.
Itemset: a group of items purchased together in a single transaction.

How Apriori Works:

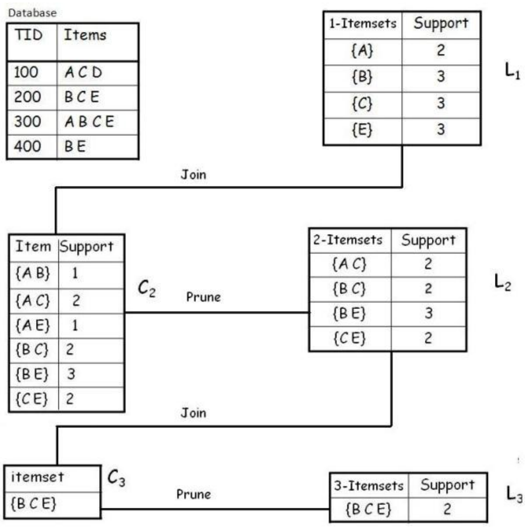
- ✓ Find all frequent itemsets:
- ✓ Get frequent items:
- ✓ Items whose occurrence in database is greater than or equal to the min.support threshold.
- ✓ Get frequent itemsets:
- ✓ Generate candidates from frequent items.
- ✓ Prune the results to find the frequent itemsets.
- ✓ Generate strong association rules from frequent itemsets
- ✓ Rules which satisfy the min.support and min.confidence threshold.

Apriori Explanation:
A database has five transactions. Let the min sup = 50% and min con f = 80%.

Database

TID	Items
100	A C D
200	B C E
300	A B C E
400	B E

Solution
Step 1: Find all Frequent Itemsets



Frequent Itemsets
{A} {B} {C} {E} {A C} {B C} {B E} {C E} {B C E}

Step 2: Generate strong association rules from the frequent itemsets

Rules	Support (X Y)	Support(X)	Confidence
{A} -> {C}	2	2	100
{B} -> {C}	2	3	66.66666667
{B} -> {E}	3	3	100
{C} -> {E}	2	3	66.66666667
{B} -> {C E}	2	3	66.66666667
{C} -> {B E}	2	3	66.66666667
{E} -> {B C}	2	3	66.66666667
{C} -> {A}	2	3	66.66666667
{C} -> {B}	2	3	66.66666667
{E} -> {B}	3	3	100
{E} -> {C}	2	3	66.66666667
{C E} -> {B}	2	2	100
{B E} -> {C}	2	3	66.66666667
{B C} -> {E}	2	2	100

IV. RESULT

The performance of the algorithms on the Foodmart dataset is shown in Fig. 16. Results show that AprioriHC-D runs faster than both Two-Phase and AprioriHC. The execution time of AprioriHC and AprioriHC-D is similar in Phase II. When min utility ¼ 0:05%, AprioriHC-D is three times faster than Two-Phase in total execution time. Table 7 shows the number of candidates generated by Two-Phase, AprioriHC and AprioriHC-D. AprioriHC and AprioriHC-D generate much less candidates than Two-Phase. For example, when min utility ¼ 0:05%, Two-Phase generates 23318 candidates, and AprioriHC and AprioriHC-D generate both 6,657 candidates. The reason is that AprioriHC and AprioriHC-D produce candidates for CHUIs, but Two-Phase needs to produce candidates for all HUIs.

In the total execution time of UP-Growth is less than CHUD, initially. But as the min_utility threshold became smaller, CHUD becomes faster (up to twice faster than UP-Growth). The reason why the performance gap between CHUD and UP-Growth is smaller for Foodmart than for Mushroom is due to the fact that Foodmart is a sparse dataset. As a consequence the reduction achieved by mining CHUIs is less (still up to 34.6 (230,610/6,656) times, as shown in Table 8). Note that achieving a smaller reduction for sparse datasets is a well-known phenomenon in frequent closed itemset mining. A similar phenomenon occurs in closedþ high utility itemset mining. Besides, when DAHU was combined with CHUD, the execution time of CHUD & DAHU was up to twice faster than UP-Growth and slightly slower than CHUD.

V. Conclusion

In this segment, we compress the exploratory results and think about qualities of the diverse calculations. To start with, we think about AprioriHC and AprioriHC-D. By and large, AprioriHC- D runs speedier and produces less hopefuls than AprioriHC. This is on the grounds that AprioriHC-D applies DGU and IIDS to prune applicants and it computes the careful utilities of applicants by examining the trimmed database rather than the first database. For thick datasets, for example, Foodmart, AprioriHC-D performs superior to anything AprioriHC when min_utility is high on the grounds that there are numerous unpromising things and secluded things when min_utility is high. In the tests, both calculations can't perform well on thick databases when min_utility is low since they experience the ill effects of the issue of a lot of hopefuls.

The most proficient calculation is CHUD. The general execution time of CHUD is constantly quicker than UP-Growth, particularly at the point when there is a great deal less CHUIs than HUIs. In addition, the blend of CHUD and DAHU is likewise speedier than UP-Growth for aggregate execution time. In the examinations, determining all HUIs is economical. In spite of the fact that CHUD þ DAHU performs superior to UP-Growth, it might neglect to recoup all high utility itemsets because of the memory restriction when there are an excessive number of HUIs in the database.

Despite the fact that CHUD performs superior to the proposed two Apriori-based methodologies, the Apriori-based methodologies are less demanding to be fathomed and actualized by the per users. Also, for a few applications, for example, finding designs within the sight of the memory limitation [5], Apriori-based methodology is favoured and plays a vital part. Despite the fact that AprioriHC and AprioriHC-D don't perform superior to anything CHUD sometimes, they are very broad and simple to be stretched out for more applications. Contingent upon the attributes of datasets, the decrease proportions accomplished by the proposed representation can be exceptionally distinctive. For the thick dataset, for example, Mushroom, the proposed representation can accomplish a huge decrease in the number of extricated examples. For the scanty dataset, for example, Foodmart, the proposed representation accomplishes less decrease.

For the dataset containing long exchanges such as BMSWebView1, an enormous lessening can be accomplished by the proposed representation. In spite of the fact that the proposed representation may not accomplish a huge decrease on exceptionally inadequate datasets, despite everything it has great execution in a few genuine cases.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. of the 20th Int'l Conf. on Very Large Data Bases, pp. 487-499, 1994.

[2] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee. Efficient Tree Structures for High utility Pattern Mining in Incremental Databases. In IEEE Transactions on Knowledge and Data Engineering, Vol. 21, Issue 12, pp. 1708-1721, 2009.

[3] J. -F. Boulicaut, A. Bykowski, and C. Rigotti. Free-sets: a condensed representation of boolean data for the approximation of frequency queries. In Data Mining and Knowledge Discovery, Vol. 7, Issue 1, pp. 5–22.

[4] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In Proc. of the Int'l Conf. on European Conference on Principles of Data Mining and Knowledge Discovery, pp. 74-85, 2002.

[5] C.-J. Chu, V. S. Tseng, T and Liang. An efficient algorithm for mining temporal high utility itemsets from data streams. In Journal of Systems and Software Vol. 81, Issure 7, pp. 1105-1117, 2008.

[6] C.-J. Chu, V. S. Tseng, and T. Liang. An Efficient Algorithm for Mining High utility Itemsets with Negative Values in Large Databases. In Applied

[7] Mathematics and Computation, Vol. 215, Issue. 2, pp. 767-778, 2009.

[8] R. Chan, Q. Yang, and Y. Shen. Mining high utility itemsets. In Proc. of IEEE Int'l Conf. on Data Mining, pp. 19-26, 2003.

[9] A. Erwin, R. P. Gopalan, and N. R. Achuthan. Efficient Mining of High utility Itemsets from Large Datasets. In Int'l Conf. on PAKDD, pp. 554-561, 2008.

[10] K. Gouda and M. J. Zaki. Efficiently mining maximal frequent itemsets. In Proc. of IEEE Int'l Conf. on Data Mining, pp. 163-170, 2001.

[11] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In Proc. of the ACM SIGMOD Int'l Conf. on Management of Data, pp. 1-12, 2000.

[12] H.-F. Li, H.-Y. Huang, Y.-C. Chen, Y.-J. Liu and S.-Y. Lee. Fast and Memory Efficient Mining of High

Utility Itemsets in Data Streams. In Proc. of IEEE Int'l Conf. on Data Mining, pp. 881-886, 2008.

[13] B. Le, H. Nguyen, T. A. Cao, and B. Vo. A Novel Algorithm for Mining High utility Itemsets. In Proc. of First Asian Conference on Intelligent Information and Database Systems, pp.13-17, 2009.

[14] Y. Liu, W. Liao, and A. Choudhary. A fast high utility itemsets mining algorithm. In Proc. of the Utility-Based Data Mining Workshop, pp. 90-99, 2005.

[15] C. Lucchese, S. Orlando and R. Perego, "Fast and Memory Efficient Mining of Frequent Closed Itemsets," In IEEE Transactions on Knowledge and Data Engineering, Vol. 18, Issue 1, pp. 21-36, 2006.

[16] Y.-C. Li, J.-S. Yeh, and C.-C. Chang. Isolated Items Discarding Strategy for Discovering High utility Itemsets. In Data & Knowledge Engineering, Vol. 64, Issue 1, pp. 198-217, 2008.

[17] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattice," In Journal of Information Systems, Vol 24, Issue 1, pp. 25–46, 1999.

[18] N. Pasquier, T. Bastide, R. Taouil, and L. Lakhal. Discovering Frequent Closed Itemsets for Association Rules. In Proc. of Int'l Conf. on Databases Theory, pp. 398–416, Israel, 1999.

[19] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Generating a Condensed representation for Association Rules, In Journal of Intelligent Information Systems, Vol 24, Issue 1, pp. 29–60, 2005.

[20] B.-E. Shie, V. S. Tseng, and P. S. Yu. Online Mining of Temporal Maximal Utility Itemsets from Data Streams. In Proc. of Annual ACM Symposium on Applied Computing, pp. 1622-1626, 2010.

[21] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu. UP-Growth: an efficient algorithm for high utility itemset mining. In Proc. of Int'l Conf. on ACM SIGKDD, pp. 253–262, 2010.

[22] B. Vo, H. Nguyen, T. B. Ho, and B. Le. Parallel Method for Mining High utility Itemsets from Vertically Partitioned Distributed Databases. In Proc. of Int'l Conf. on Knowledge-based and Intelligent Information and Enginnering Systems, pp. 251-260, 2009.

[23] J. Wang, J. Han, and J. Pei. Closet+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In Proc. of Int'l Conf. on ACM SIGKDD, pp. 236–245, 2003.

[24] U. Yun. Mining lossless closed frequent patterns with weight constraints. In Knowledge-Based Systems, Vol. 20, pp. 86–97, 2007.

[25] J.-S. Yeh, C.-Y. Chang, and Y.-T. Wang. Efficient Algorithms for Incremental Utility Mining. In Proc. of the 2nd Int'l Conf. on Ubiquitous information management and communication, pp. 212-217, 2008.

[26] H. Yao, H. J. Hamilton, L. Geng, A unified framework for utility-based measures for mining itemsets. In Proc. of ACM SIGKDD 2nd Workshop on Utility-Based Data Mining, pp. 28-37, 2006.

[27] S.-J. Yen and Y.-S. Lee. Mining High utility Quantitative Association Rules. In Proc. of Int'l Conf. on Data Warehousing and Knowledge Discovery, pp. 283-292, 2007.

[28] M. J. Zaki and C. J. Hsiao. Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. In IEEE Transactions on Knowledge and Data Engineering, Vol. 17, Issue 4, pp. 462–478, 2005. Frequent itemset mining implementations repository, http://fimi.cs.helsinki.fi/