Mining of High Average-Utility Patterns with Item-Level Thresholds

Jerry Chun-Wei Lin[1], Ting Li[2], Philippe Fournier-Viger[3], Ji Zhang[4], Xiangmin Guo[5]

[1] Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences, Bergen, Norway
[2] School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China
[3] School of Humanities and Social Sciences, Harbin Institute of Technology (Shenzhen), China
[4] School of Agricultural, Computational and Environmental Sciences, University of Southern Queensland, Australia
[5] School of Architecture and Urban Planning, Harbin Institute of Technology (Shenzhen), China
jerrylin@ieee.org, tingli@ikelab.net, philfv@hitsz.edu.cn, ji.zhang@usq.edu.au, xiangminguo@gmail.com

Abstract

In this paper, we introduce a level-wise algorithm named High Average-Utility Itemset Mining with Multiple Minimum Average-Utility threshold (HAUIM-MMAU), which relies on a novel transaction-maximum utility downward closure (TMUDC) property and a concept of least minimum average-utility (LMAU) to mine high average-utility itemsets (HAUIs). Two efficient strategies, named IEUCP and PBCS, are designed to further reduce the search space, and thus speed up the performance of HAUI mining. Several experiments carried out on both synthetic and real-life databases show that the proposed algorithm can efficiently discover the complete set of HAUIs while considering multiple minimum average-utility thresholds.

Keywords: Average-utility itemsets, Multiple thresholds, Transaction-maximum utility downward closure, IEUCP, PBCS

1 Introduction

Due to the emerging growth of sensor network [6, 28] and Internet of Things (IOT), it is necessary to find the useful and meaningful information from a very large dataset. The main purpose of Knowledge Discovery in Databases (KDD) is to discover implicit and useful information in a collection of data. Association Rule Mining (ARM) [1-2, 4, 9] is an important topic in KDD, which has been extensively studied. Frequent Itemset Mining (FIM) is a fundamental task in ARM, which aims at finding groups of items (itemsets) that frequently appear in a transactional database. However, traditional algorithms for these tasks may find many patterns that are frequent but uninteresting to the user, and also miss many rare but interesting patterns.

To address this limitation, the problem of high utility itemset mining (HUIM) [5, 13, 15, 22-24, 27, 29-32] was introduced. An important limitation of traditional HUIM is that the utility of an itemset is generally smaller than the utility of its supersets. To reduce the influence of an itemset's length on its utility, and find more useful high utility itemsets for recommendation, Hong et al. proposed the average utility measure, and the problem of High Average Utility Itemset Mining (HAUIM) [11]. To discover all high average-utility itemsets (HAUIs) in a database efficiently, the average-utility upper-bound auub model [11] was designed to restore the Downward Closure (DC) property. Numerous algorithms have been designed to more efficiently mine HAUIs [19-21]. However, most of them rely on a single minimum average-utility threshold to mine HAUIs. It is thus unfair to use a single threshold to assess the utility of all patterns in a database.

To address this issue, this paper proposes a novel framework for high average-utility itemset mining with multiple minimum average-utility thresholds (HAUIM-MMAU). The key contributions of this paper are fourfold:

1. This paper proposes a framework for discovering HAUIs by considering multiple minimum average-utility thresholds. In this framework, the user can set a distinct minimum average-utility threshold for each item instead of using a single minimum average-utility threshold for all items.

2. Based on the proposed framework, a two-phase algorithm named HAUI-MMAU is proposed to discover HAUIs. It relies on the novel concept of Least Minimum Average Utility (LMAU) and a designed Transaction-Maximum-Utility Downward Closure (TMUDC) property, to mine HAUIs.

3. To improve the performance of the proposed algorithm, two efficient pruning strategies called IEUCP and PBCS are designed. These strategies are developed to prune unpromising itemsets early, thus

*Corresponding Author: Jerry Chun-Wei Lin; E-mail: jerrylin@ieee.org
DOI: 10.3966/160792642019012001017

reducing the search space and speeding up the discovery of HAUIs.

2 Literature Review

In recent years, HUIM [5, 15, 27, 32] has become a key research topic in the field of data mining. The key difference is that, in HUIM each item may be associated with a distinct weight to indicate its relative importance, and each item may have non binary purchase quantities in transactions. Chan et al. [5] proposed the idea of utility mining. They designed an approach that discovers both frequent itemsets and HUIs. Yao et al. [32] then proposed a strict unified framework of utility mining that considers both purchase quantities of items in transactions (internal utilities) and their unit profits (external utilities).

A key problem with the above approaches is that the utility measure does not follow the Downward Closure (DC) property. In other words, supersets of a high utility itemset may be low utility itemsets or high utility itemsets. To address this problem, Liu et al. [15] introduced the Transaction-Weighted Utility (TWU) model and the Transaction-Weighted Downward Closure (TWDC) property. Liu et al. [18] proposed the HUI-Miner algorithm to discover HUIs without generating candidates using a designed utility-list structure. Fournier-Viger et al. [7] presented the FHM algorithm and the Estimated Utility Co-occurrence Structure (EUCS) to mine HUIs.

An important drawback of HUIM is that it does not consider the length of itemsets. To reduce the influence of an itemset's length on its utility, and thus find more useful high utility itemsets, Hong et al. proposed the average utility measure, and the problem of high average utility itemset mining (HAUIM) [11]. Hong et al. [11] first designed the Two-Phase Average-Utility algorithm (TPAU) to mine HAUIs. Lin et al. [19] then developed a high average-utility pattern (HAUP)-tree structure and adopted the auub model, to mine HAUIs more efficiently. Lan et al. [20] developed a projection-based average-utility (PBAU) mining algorithm to mine HAUIs. Lu et al. [21] then developed a HAUI-tree structure to mine HAUIs without candidate generation. Lin et al. [25] presented a more efficient algorithm to mine HAUIs. Lin et al. [26] also designed two models called the looser upper-bound model (lub) and the upper-bound (rtub) model to ignore irrelevant items in transactions to further tighten the upper-bounds. The above algorithms were designed to mine HAUIs using a single minimum average-utility threshold (count). But in real-life situations, items are often regarded as having different importance to the user. Hence, it is unfair to assess the utility of all items using a same minimum average utility threshold (count), to determine if itemsets are HAUIs.

In FIM and ARM, each item also has its own weight or importance. Thus, it is unfair to mine associations rules (ARs) or frequent itemsets (FIs) using a same minimum support threshold. The first proposed algorithm to mine FIs while considering multiple minimum support thresholds is MSApriori [14]. MSApriori explores the search space using a level-wise approach. In another study, the CFP-growth algorithm [10] was designed. It defines a MIN value as the smallest MIS value among all items in the database. Kiran and Reddy then presented the CFP-growth++ algorithm [12] to identify the items that cannot generate any frequent pattern. The MHU-Growth algorithm [29] extends CFP-Growth to mine high utility frequent itemsets with multiple minimum support thresholds. It first constructs a MHU-Tree structure that stores the supports and TWU values of items in tree nodes. Lee et al. [16] presented the maximum constraint approach to mine frequent itemsets with multiple minimum support thresholds. A similar idea was proposed for fuzzy data mining with multuple levels [17]. The proposed algorithm adopts a top-down progressively deepening approach to derive frequent itemsets, and extract cross-level fuzzy association rules under the maximum-itemset minimum-taxonomy support constraint.

3 Preliminaries and Problem Statement

Let $I = \{i_1, i_2, \ldots, i_r\}$ be a finite set of r distinct items occurring in a database D, and $D = \{T_1, T_2, \ldots, T_n\}$ be a set of transactions, where for each transaction $T_q \in D$, T_q is a subset of I and has a unique identifier q, called its *TID* (Transaction IDentifier). Moreover, a profit table $ptable = \{p_{i1}, p_{i2}, \ldots, p_{ir}\}$ is defined, where p_{im} is a positive integer representing the unit profit of item i_m ($1 \le m \le r$). A set of k distinct items $X = \{i_1, i_2, \ldots, i_k\}$ such that $X \subseteq I$ is said to be a k-itemset, where k is the length or level of the itemset. An itemset X is said to be contained in a transaction T_q if $X \subseteq T_q$.

An example quantitative database is shown in Table 1. It consists of five transactions and six items, represented using letters from (*a*) to (*f*). A profit table is defined as {*a*:5, *b*:2, *c*:1, *d*:2, *e*:4, *f*:1}, which is the profit value of each item. Henceforth, this database will be used as the running example.

Table 1. A quantitative database

TID	Items
1	*b*:7, *c*:2, *d*:3, *e*:1
2	*b*:4, *c*:3, *d*:3
3	*a*:2, *d*:1
4	a:1, c:6, f :4
5	b:2, c:3, d:1, f :2

Definition 1. The minimum average-utility threshold to be used for an item i_j in a database D is a positive integer denoted as $mau(i_j)$. A MMAU-table is used to store the minimum average-utility thresholds of all

items in D, and is defined as:

Definition 2. For a k-itemset X, the minimum average-utility threshold of X is denoted as $mau(X)$, and is defined as:

$$mau(X) = \frac{\sum_{i_j \in X} mau(i_j)}{|X|} = \frac{\sum_{i_j \in X} mau(i_j)}{k}.$$

Definition 3. The average-utility of an item i_j in a transaction T_q is denoted as $au(i_j, T_q)$, and is defined as:

$$au(i_j, T_q) = \frac{q(i_j, T_q) \times p(i_j)}{1}.$$

Definition 4. The average-utility of a k-itemset X in a transaction T_q is denoted as $au(X, T_q)$, and defined as:

$$au(X, T_q) = \frac{\sum_{i_j \in X \subseteq T_q} q(i_j, T_q) \times p(i_j)}{|X| = k},$$

where k is the number of items in X.

Definition 5. The average-utility of an itemset X in a database D is denoted as $au(X)$, and is defined as:

$$au(X) = \sum_{X \subseteq T_q \in D} au(X, T_q).$$

Definition 6. An itemset X in a database D is said to be a HAUI if its average utility $au(X)$ is no less than its minimum average-utility threshold $mau(X)$, that is:

$$HAUI \leftarrow \{X | au(X) \geq mau(X)\}.$$

Problem Statement: The purpose of HAUIM-MMAU is to efficiently discover the set of all HAUIs, where an itemset X is said to be a HAUI if its average utility is no less than its minimum average-utility threshold $mau(X)$.

4 Proposed Algorithm and Strategies

Based on the proposed HAUIM-MMAU framework, a baseline HAUI-MMAU algorithm is proposed in this section. Thereafter, two improved algorithms based on two novel pruning strategies named IEUCP and PBCS are introduced to further increase its efficiency.

4.1 Proposed Downward Closure Property

To speed up the discovery of frequent itemsets, the well-known Downward Closure (DC) property is used in FIM to prune unpromising itemsets early, and thus reduce the search space. This model allows pruning the search space early for discovering HAUIs.

Definition 6. An itemset X is a high average-utility upper-bound itemset (HAUUBI) if its $auub$ is no less than its minimum average-utility threshold. The set of all HAUUBIs is thus defined as:

$$\text{HAUUBI} \leftarrow \{auub(X) \geq mau(X)\}.$$

Definition 7. The least minimum average-utility ($LMAU$) in the $MMAU$-$table$ is defined as:

$$LMAU = min\{mau(i_1), mau(i_2), \ldots, mau(i_r)\},$$

where r is the total number of items.

Strategy 1. The items in the MMAU-table are sorted in ascending order of their mau values.

Based on the designed strategy 1, the following transaction-maximum-utility downward closure (TMUDC) property holds for the HAUIM-MMAU framework.

Theorem 1. (Transaction-maximum-utility downward closure property, TMUDC property). Without loss of generality, assume that items in itemsets are sorted in ascending order of mau values. Let X^k be a k-itemset ($k \geq 2$), and X^{k-1} be a subset of X^k of length k-1. If X^k is a HAUUBI, then X^{k-1} is also a HAUUBI.

Besides, from the $auub$ model, we can ensure that HAUIs $\subseteq$ HAUUBIs. By using this approach for candidate generation, many unpromising candidates can be eliminated, and it becomes unnecessary to calculate their actual average-utility values in the later mining process.

4.2 Proposed HAUI-MMAU Algorithm

In the first phase, the designed HAUI-MMAU algorithm performs a breadth-first search to mine the HAUUBIs. The pseudo-code of the proposed algorithm is given in Algorithm 1.

Algorithm 1. HAUI-MMAU

Input: D, a quantitative transactional databases; $ptable$, a profit table; $MMAU$-$table$, the user pre-defined multiple minimum average utility threshold table.

Output: The complete set of high average-utility itemsets (HAUIs).

1. find the $LMAU$ in the $MMAU$-$table$;
2. scan D to find $auub(i_j)$;
3. **for** each item i_j **do**
4. **if** $auub(i_j) \geq LMAU$ **then**
5. $\text{HAUUBI}^1 \leftarrow \text{HAUUBI}^1 \cup i_j$;
6. sort items in HAUUBI^1 in ascending order of their mau values;
7. set $k = 2$;
8. **while** $\text{HAUUBI}^k \neq$ ***null*** **do**
   C_k = generate_candidate(HAUUBI^{k-1});
9. **for** each k-itemset $X \in C_k$ do
10. scan D to calculate $auub(X)$;
11. **if** $auub(X) \geq mau(X)$ **then**
12. $\text{HAUUBI}^k \leftarrow \text{HAUUBI}^k \cup X$;
13. set $k = k + 1$;
14. $\text{HAUUBIs} \leftarrow \text{HAUUBI}^k$;
15. **for** each itemset X in HAUUBIs **do**
16. scan D to calculate $au(X)$;
17. **if** $au(X) \geq mau(X)$ **then**
18. $\text{HAUIs} \leftarrow \text{HAUIs} \cup X$;
19. return HAUIs;

The proposed HAUI-MMAU algorithm takes as input: (1) a quantitative transactional database D (2) a profit table *ptable* indicating the unit profit of each item, and (3) a multiple minimum average-utility threshold table, *MMAU-table*. The algorithm performs the following steps. First, the least minimum average-utility value (*LMAU*) is found in the *MMAU-table* (Line 1). Then, the database is scanned to find the *auub* values of all 1-items (Line 2). For each 1-item, the algorithm checks if its *auub* value is no less than the LMAU, to determine if it is an HAUUBI of length 1 (Lines 3 to 5). After obtaining the set of all HAUUBIs of length 1 ($HAUUBI^1$), the items in $HAUUBI^1$ are sorted in ascending order of their *mau* values (Line 6). This process is necessary to ensure that all HAUIs will be retrieved, as explained in the previous section. The parameter k is then set to 2 (Line 7), and a loop is performed to mine all HAUUBIs in a level-wise way, starting from itemsets of length $k = 2$ (Lines 8 to 14). During the (k-1)-th iteration of the loop, the set $HAUUBI^k$, which contains the HAUUBIs of length k, is obtained by the following process. First, pairs of (k-1)-itemsets in $HAUUBI^{k-1}$ are combined to generate their supersets of length k by applying the generate_candidate procedure (Line 9). The result is a set named C_k, containing potential HAUUBIs of length k. The database is then scanned again to calculate the *auub* value of each itemset X in C_k. If the itemset X has an *auub* value that is no less than its maximum average-utility (*mau*), it is added to the set $HAUUBI^k$ (Lines 11 to 13). This loop terminates when no more candidates can be generated.

In the second phase, the database is scanned again to find the HAUIs from the set of HAUUBIs (Lines 16 to 19). Based on the proposed theorems, the designed algorithm is **correct** and **complete**, and thus returns the full set of HAUIs (Line 20).

4.3 Improved IEUCP Strategy

The TMUDC property proposed in the designed HAUI-MMAU algorithm can considerably reduce the search space and speed up the discovery of HAUIs. However, the designed HAUI-MMAU algorithm still suffers from the combinational explosion of the number of itemsets in the search space. To address this limitation, the estimated utility co-occurrence pruning (EUCP) strategy introduced in FHM [7] is adapted for the problem of MAUIM-MMAU. Since the TWDC property used in the TWU model does not hold in the designed HAUIM-MMAU framework, the original EUCP strategy used in FHM cannot be directly applied to the proposed algorithm. This issue is addressed by using the developed TMUDC property instead of the TWDC. The resulting strategy is named the Improved Estimated Utility Co-occurrence Pruning strategy (IEUCP). This strategy aims to reduce the number of join operations (the operation of combining pairs of itemsets to generate larger itemsets) by pruning unpromising itemsets early. The IEUCP strategy is applied for the generation of k-itemsets ($k \geq 2$). It relies on the following theorem to ensure the correctness and the completeness of the HAUI-MMAU algorithm with the IEUCP strategy, for discovering HAUIs.

Theorem 2. (Improved Estimated Utility Co-occurrence Pruning strategy, IEUCP). Let there be an itemset X^{k-1} and an itemset X^k that is an extension of X^{k-1} (that was obtained by appending an item to X^{k-1}). Without loss of generality, assume that items in itemsets are sorted by ascending order of *mau* values. A sorted prefix of X^{k-m} is a subset of X^{k-1} containing the m first items of X^{k-1} according to the ascending order of *mau* values, where $1 \leq m < k$-1. For example, consider the itemset (*bcde*), which is a 4-itemset ($k = 4$). Its sorted prefixes are (*b*), (*bc*), and (*bcd*). If there exists a sorted prefix X^{k-m} of X^k that is not a HAUUBI, then X^k is also not a HAUUBI.

It is important to note that an itemset X can only be safely pruned by Theorem 2 if X has a sorted prefix that is not a HAUUBI, and that this pruning condition should not be applied by considering subsets of X that are not sorted prefixes of X. A structure for storing the *auub* values of 2-itemsets is further used by the IEUCP strategy, and is shown in Table 2. This structure is efficiently constructed during initial database scans.

Table 2. The 2-itemsets structure of IEUCP

Item	*a*	*b*	*f*	*c*	*d*	*e*
a	-	-	-	-	-	-
b	0	-	-	-	-	-
f	6	4	-	-		
c	6	26	10	-	-	-
d	10	26	4	26	-	-
e	0	14	0	14	14	-

4.4 Pruning Before Calculation Strategy, PBCS

Although the developed IEUCP strategy is effective at reducing the number of unpromising candidates in the first phase, calculating the set of actual HAUIs in the second phase remains a very time-consuming process. To address this issue, this article further proposes an efficient pruning before calculation (PBCS) strategy to prune itemsets without performing a database scan.

Theorem 3. Let X, X^a, and X^b be itemsets such that $X^a \subset X$, $X^b \subset X$, and $X^a \cup X^b = X$, $X^a \cap X^b = null$. If both X^a and X^b are not HAUIs, the itemset X is also not a HAUI.

Thus, if it is found that two subsets X^a and X^b of an itemset X are not HAUIs and respect the condition of Theorem 3, it can be concluded that X is not a HAUI. It is thus unnecessary to scan the database to calculate the actual utility of X. Based on this PBCS strategy, the cost of calculating the actual utility of HAUUBIs in the second phase can be greatly reduced.

5 Experimental Evaluation

Substantial experiments were conducted to evaluate the effectiveness and efficiency of the proposed algorithms. The performance of the baseline HAUI-MMAU algorithm was compared with two versions named HAUI-MMAU$_{IEUCP}$ and HAUI-MMAU$_{PBCS}$, which respectively integrates the proposed IEUCP strategy, and both the designed IEUCP and PBCS strategies. All algorithms were implemented in Java. Experiments were carried on both real-life and synthetic datasets, having varied characteristics. The foodmart, retail, kosarak and chess datasets were obtained from the SPMF website [8]. Furthermore, the method proposed in [14] for assigning the multiple thresholds to items was adapted to automatically set the *mau* value of each item in the proposed HAUIM-MMAU framework. The following equation is thus used to set the *mau* value of each item i_j:

$$mau(i_j) = max\{\beta \times p(i_j), GLMAU\},$$

where β is a constant used to set the *mau* values of an item as a function of its unit profit. To ensure randomness and diversity in the experiments, β was set in the [1, 1000] interval for retail and the T10I4D100K datasets, [1, 10] for the foodmart dataset, and [10k, 15k] for the T40I10D100K dataset. The constant *GLMAU* is user-specified and represents the global least average-utility value. Lastly, $p(i_j)$ represents the external utility (unit profit) of the item i_j. If β is set to zero, a single minimum average-utility threshold GLMAU is used for all items. In that case, the task of HAUIM-MMAU becomes that of traditional HAUIM.

5.1 Runtime

In this section, the runtime of the proposed HAUI-MMAU algorithm and the two improved algorithms named HAUI-MMAU$_{IEUCP}$ and HAUI-MMAU$_{PBCS}$, are respectively compared. Figure 2 shows the runtime of the proposed algorithms when β is varied within the predefined interval for each dataset and the *GLMAU* value is fixed for the six datasets.

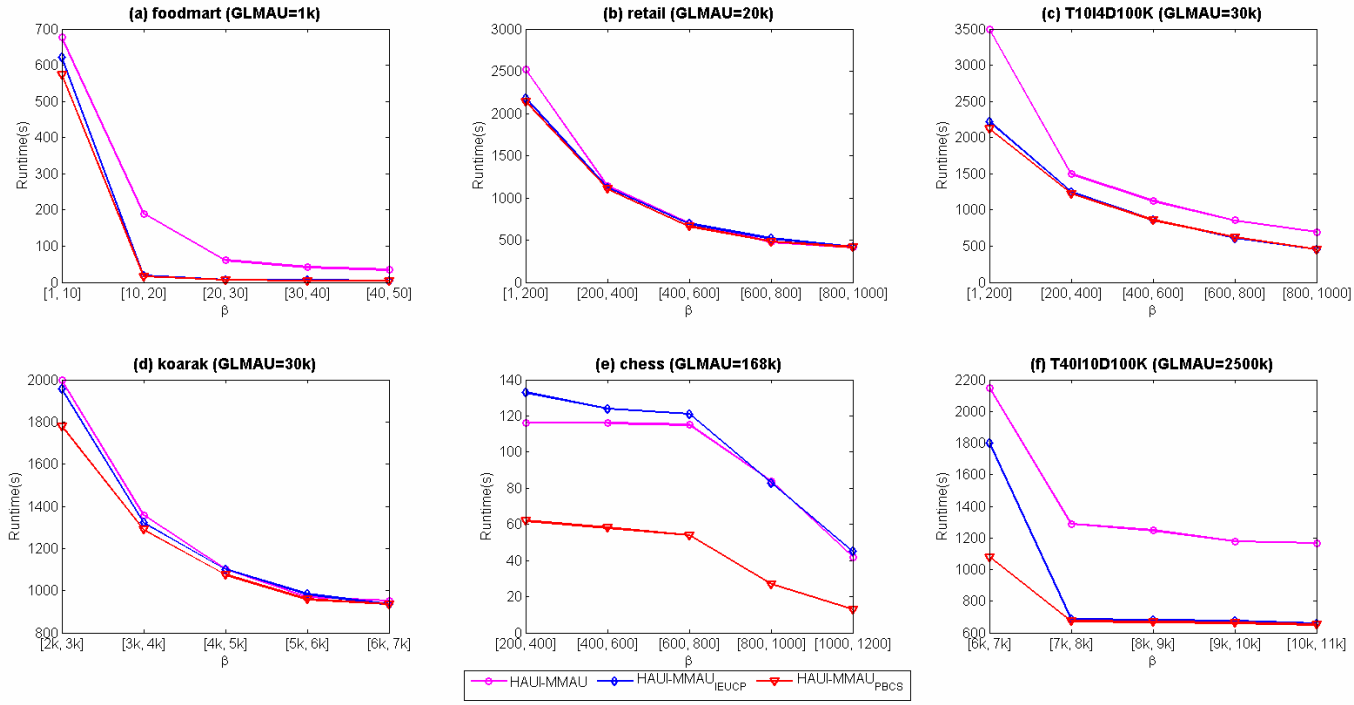

Figure 2. Runtimes for a fixed *GLMAU* value while varying β

It can be observed in Figure 2 that the improved HAUI-MMAU$_{IEUCP}$ and HUAI-MMAU$_{PBCS}$ algorithms outperform the baseline HAUI-MMAU algorithm for the six datasets for a fixed *GLMAU* and various values of β. It can also be observed that HAUI-MMAU$_{PBCS}$ always has the best performance among the three algorithms. The reason is that HAUI-MMAU$_{PBCS}$ applies both the IEUCP and PBCS strategies to improve the performance of the mining process, while HAUI-MMAUIEUCP only applies the IEUCP strategy. But both strategies are complementary to each other. The IEUCP strategy can prune a large amount of unpromising itemsets to avoid several database scans when calculating the *auub* values of itemsets in the first phase, while the PBCS strategy can prune unpromising HAUIs from the remaining HAUUBIs to avoid calculating their actual average-utilities, in the second phase. Hence, HAUI-MMAU$_{PBCS}$ has the best performance among the three algorithms.

5.2 Candidate Analytics

We further compared the number of candidates generated by each of the three algorithms. Here, an itemset is said to be a candidate if its actual average-

utility value is calculated in the first phase, and its exact average-utility is calculated in the second phase. Figure 3 shows the number of candidates for the three algorithms when β is varied within the predefined interval for each dataset, and with a fixed *GLMAU* value.

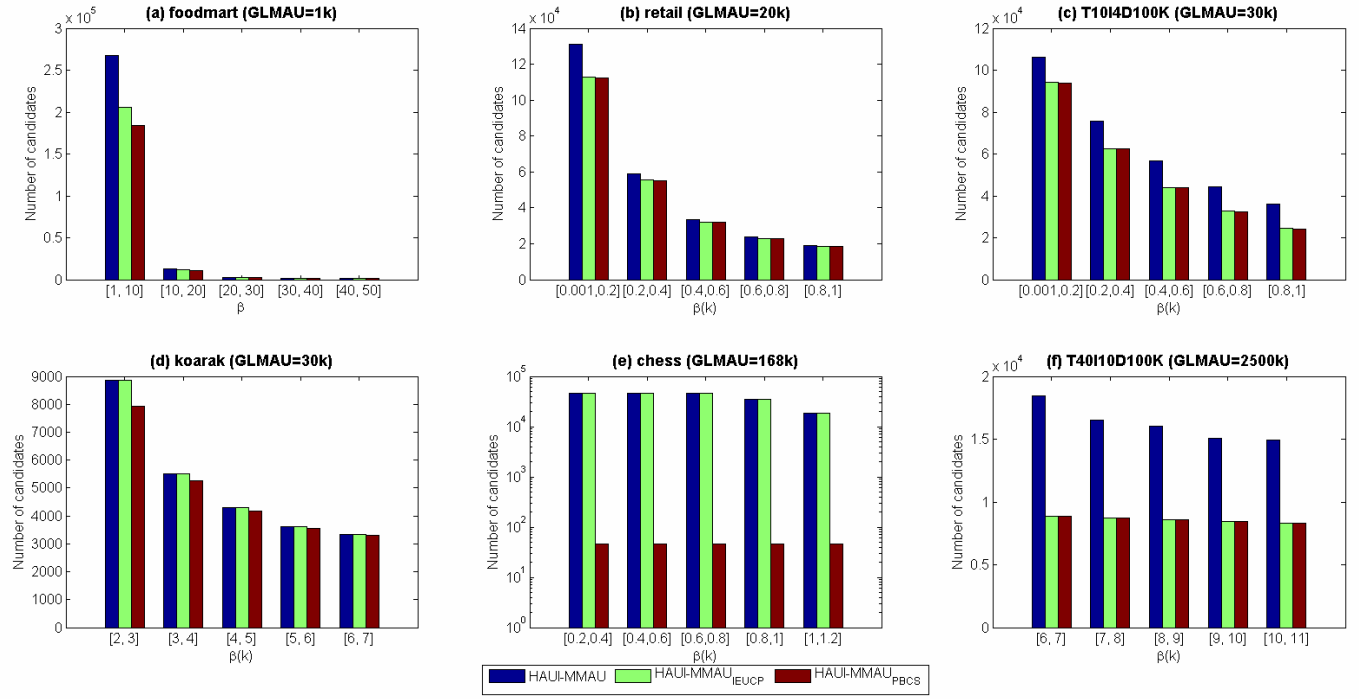

Figure 3. Number of candidates for a fixed *GLMAU* while varying β

In Figure 3, it can be observed that the HAUI-MMAU algorithm always consider more candidates than the other two algorithms. The reason is that HAUI-MMAUIEUCP can efficiently prune some unpromising HAUUBIs using its designed pruning strategy, while HAUI-MMAUPBCS can prune unpromising HAUIs from the remaining HAUUBIs for revealing the actual HAUIs. It can be observed in Figure 3(c) and Figure 3(d), that the number of candidates considered by HAUI-MMAUIEUCP and HAUI-MMAUPBCS is almost the same. This explains why these algorithms have similar runtimes, as shown in Figure 3(c) and Figure 3(d). But in Figure 3(e), it is obvious that the number candidates considered by HAUI-MMAUPBCS is much less than the two other algorithms, which shows that the proposed PBCS strategy can greatly reduce the search space in the second phase for the chess dataset.

6 Conclusion

In this paper, the HAUIM-MMAU framework was designed to mine HAUIs with multiple minimum average-utility thresholds. An extensive experimental study was conducted on both synthetic and real datasets to evaluate the performance of the algorithms in terms of runtime and number of candidates.

Acknowledgments

This research was partially supported by National Natural Science Foundation of China (NSFC) under grant No.61503092, by the Shenzhen Technical Project under JCYJ20170307151733005, and by the Science Research Project of Guangdong Province under grant 2017A020220011.

References

[1] R. Agrawal, T. Imielinski, A. Swami, Database Mining: A Performance Perspective, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, pp. 914-925, December, 1993.

[2] R. Agrawal, R. Srikant, Fast Algorithms for Mining Association Rules, *International Conference on Very Large Data Bases*, Santiago, Chile, 1994, pp. 487-499.

[3] R. Agrawal, R. Srikant, Quest Synthetic Data Generator, 2017, https://sourceforge.net/projects/ibmquestdatagen.

[4] M. S. Chen, J. Han, P. S. Yu, Data Mining: An Overview from Database Perspective, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, No. 6, pp. 866-883, December, 1996.

[5] R. Chan, Q. Yang, Y. D. Shen, Mining High Utility Itemsets, *IEEE International Conference on Data Mining*, Melbourne, FL, pp. 19-26, 2003.

[6] C. Y. Chen, H. C. Chao, A Survey of Key Distribution in Wireless Sensor Networks, *Security and Communication Networks*, Vol. 7, No. 12, pp. 2495-2508, July, 2014.

[7] P. Fournier-Viger, C. W. Wu, S. Zida, V. S. Tseng, FHM: Faster High-utility Itemset Mining Using Estimated Utility Co-occurrence Pruning, *International Symposium on Methodologies for Intelligent Systems*, Roskilde, Denmark, 2014, pp. 83-92.

[8] P. Fournier-Viger, J. C. W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, H. T. Lam, The SPMF Open-source Data Mining Library Version 2, *The Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Riva del Garda, Italy, 2016, pp. 36-40.

[9] J. Han, J. Pei, Y. Yin, R. Mao, Mining Frequent Patterns without Candidate Generation: A Frequent-pattern Tree Approach, *Data Mining and Knowledge Discovery*, Vol. 8, No. 1, pp. 53-87, January, 2004.

[10] Y. H. Hu, Y. L. Chen, Mining Association Rules with Multiple Minimum Supports: A New Mining Algorithm and a Support Tuning Mechanism, *Decision Support Systems*, Vol. 42, No. 1, pp. 1-24, October, 2006.

[11] T. P. Hong, C. H. Lee, S. L. Wang, Effective Utility Mining with the Measure of Average Utility, *Expert Systems with Applications*, Vol. 38, No. 7, pp. 8259-8265, July, 2011.

[12] R. Uday Kiran, P. Krishna Reddy, Novel Techniques to Reduce Search Space in Multiple Minimum Supports-based Frequent Pattern Mining Algorithms, *ACM International Conference on Extending Database Technology*, Uppsala, Sweden, 2011, pp. 11-20.

[13] S. Krishnamoorthy, Pruning Strategies for Mining High Utility Itemsets, *Expert Systems with Applications*, Vol. 42, No. 5, pp. 2371-2381, April, 2015.

[14] B. Liu, W. Hsu, Y. Ma, Mining Association Rules with Multiple Minimum Support, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, 1999, pp. 337-341.

[15] Y. Liu, W. Liao, A. Choudhary, A Two-phase Algorithm for Fast Discovery of High Utility Itemsets, *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Hanoi, Vietnam, 2005, pp. 689-695.

[16] Y. C. Lee, T. P. Hong, W. Y. Lin, Mining Association Rules with Multiple Minimum Supports Using Maximum Constraints, *International Journal of Approximate Reasoning*, Vol. 40, pp. 44-54, July, 2005.

[17] Y. C. Lee, T. P. Hong, T. C. Wang, Multi-level Fuzzy Mining with Multiple Minimum Supports, *Expert Systems with Applications*, Vol. 34, No. 1, pp. 459-468, January, 2008.

[18] M. Liu, J. Qu, Mining High Utility Itemsets without Candidate Generation, *ACM International Conference on Information and Knowledge Management*, Maui, HI, 2012, pp. 55-64.

[19] C. W. Lin, T. P. Hong, W. H. Lu, Efficiently Mining High Average Utility Itemsets with a Tree Structure, *Asian Conference on Intelligent Information and Database Systems*, Hue, Vietnam, 2010, pp. 131-139.

[20] G. C. Lan, T. P. Hong, V. S. Tseng, A Projection-based Approach for Discovering High Average-utility Itemsets, *Journal of Information Science and Engineering*, Vol. 28, No. 1, pp. 193-209, January, 2012.

[21] T. Lu, B. Vo, H. T. Nguyen, T. P. Hong, A New Method for Mining High Average Utility Itemsets, *IFIP International Conference on Computer Information Systems and Industrial Management*, Ho Chi Minh, Vietnam, 2014, pp. 33-42.

[22] J. C. W. Lin, W. Gan, T. P. Hong, V. S. Tseng, Efficient Algorithms for Mining Up-to-date High-utility Patterns, *Advanced Engineering Informatics*, Vol. 29, No. 3, pp. 648-661, August, 2015.

[23] J. C. W. Lin, L. Yang, P. Fournier-Viger, T. P. Hong, M. Voznak, A Binary PSO Approach to Mine High-utility Itemsets, *Soft Computing*, Vol. 21, No. 17, pp. 5103-5121, September, 2017.

[24] J. C. W. Lin, W. Gan, P. Fournier-Viger, T. P. Hong, FDHUP: Fast Algorithm for Mining Discriminative High Utility Patterns, *Knowledge and Information Systems*, Vol. 51, No. 3, pp 873-909, June, 2017.

[25] J. C. W. Lin, S. Ren, P. Fournier-Viger, T. P. Hong, J. H. Su, B. Vo, A Fast Algorithm for Mining High Average-utility Itemsets, *Applied Intelligence*, Vol. 47, No. 2, pp. 331-346, September, 2017.

[26] J. C. W. Lin, S. Ren, P. Fournier-Viger, T. P. Hong, EHAUPM: Efficient High Average-utility Pattern Mining with Tighter Upper Bounds, *IEEE Access*, Vol. 5, pp. 12927-12940, June, 2017.

[27] J. C. W. Lin, L. Yang, P. Fournier-Viger, T. P. Hong, Mining of Skyline Patterns by Considering both Frequent and Utility Constraints, *Engineering Applications of Artificial Intelligence*, Vol. 77, pp. 229-238, January, 2019.

[28] H. F. Rashvand, H. C. Chao, *Dynamic Ad-Hoc Networks*, The Institution of Engineering and Technology, 2013.

[29] H. Ryang, U. Yun, K. H. Ryu, Discovering High Utility Itemsets with Multiple Minimum Supports, *Intelligent Data Analysis*, Vol. 18, No. 6, pp. 1027-1047, November, 2014.

[30] V. S. Tseng, B. E. Shie, C. W. Wu, P. S. Yu, Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 8, pp. 1772-1786, August, 2013.

[31] V. S. Tseng, C. W. Wu, P. Fournier-Viger, P. S. Yu, Efficient Algorithms for Mining Top-K High Utility Itemsets, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 208, No. 1, pp. 54-67, January, 2016.

[32] H. Yao, H. J. Hamilton, C. J. Butz, A Foundational Approach to Mining Itemset Utilities from Databases, *SIAM International Conference on Data Mining*, Lake Buena Vista, FL, 2004, pp. 211-225.

Biographies

Jerry Chun-Wei Lin is working as the Associate Professor at Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences, Bergen, Norway. He has published more than 250 peer-reviewed papers. His research interests include big data analytics, privacy-preserving and security, soft computing, and machine learning.

Ting Li is working as the Master student in Harbin Institute of Technology (Shenzhen), China. His research interests include data mining and soft computing.

Philippe Fournier-Viger is a full professor at the Harbin Institute of Technology (Shenzhen). He has published more than 160 research papers, which have received 1,000 citations in the last two years. He is the founder of the popular SPMF data mining library (http://www.philippe-fournier-viger.com/spmf/).

Ji Zhang is currently an Associate Professor in Computing at the University of Southern Queensland, Australia. He is an Australian Endeavour Fellow, Queensland Fellow and Izaak Walton Killam Fellow (Canada). His research interests are knowledge discovery and data mining, health informatics, information privacy and security.

Xiangmin Guo is an Associate Professor of Harbin Institute of Technology Shenzhen Campus, Executive Director of OCT-HIT Future City Research Institute. His research interests include digital technology aided architecture and urban design, and urban management evaluation. He has published 3 books and more than 50 papers.