International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 1 (February '2017)
Special Issue on 1st National Conference on Advanced Computing and Communication Technologies,
Conference held at Siri PSG Arts and Science College for Women, Sankari,Tamilnadu,India.

HIGH UTILITY ITEMSETS MINING USING APRIORITY CH AND CLOSED HIGH UTILITY ITEMSET DISCOVERY

R.Kalaiselvi,
Research Scholar,
Department Of Computer Science,
Theivanai Ammal College for Women,
Villupuram,Tamilnadu,India.

Dr.R.Suguna,
Assistant Professor,
Department Of Computer Science,
Theivanai Ammal College for Women,
Villupuram,Tamilnadu,India.

Abstract: Mining high utility item sets(HUIs) from databases is an important data mining task, which refers to the discovery of itemsets with high utilities (e.g. high profits). However, it may present too many HUIs to users, which also degrades the efficiency of the mining process. To achieve high efficiency for the mining task and provide a concise mining result to users, proposed a novel framework in this paper for mining closed high utility itemsets (CHUIs), which serves as a compact and lossless representation of HUIs. Proposed three efficient algorithms named AprioritycH(Apriori-based algorithm for mining High utility closed itemsets), AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items) and CHUD (Closed High Utility Itemset Discovery) to find this representation. Further, a method called DAHU (Derive All High Utility Itemsets) is proposed to recover all HUIs from the set of CHUIs without accessing the original database. Results on real and synthetic datasets show that the proposed algorithms are very efficient and that our approaches achieve a massive reduction in the number of HUIs. In addition, when all HUIs can be recovered by DAHU, the combination of CHUD and DAHU outperforms the state-of-the-art algorithms for mining HUIs.

Keywords: Frequent item set, closed high utility item set, lossless and concise representation, utility mining

I.INTRODUCTION

Frequent item set mining (FIM) is a fundamental research topic in data mining. One of its popular applications is market basket analysis, which refers to the discovery of sets of items (itemsets) that are frequently purchased together by customers. However, in this application, the traditional model of FIM may discover a large amount of frequent but low revenue itemsets and lose the information on valuable itemsets having low selling frequencies. These problems are caused by the facts that (1) FIM treats all items as having the same importance/unit profit/weight and (2) it assumes that every item in a transaction appears in a binary form, i.e., an item can be either present or absent in a transaction, which does not indicate its purchase quantity in the transaction. Hence, FIM cannot satisfy the requirement of users who desire to discover itemsets with high utilities such as high profits. To address these issues, utility mining emerges as an important topic in data mining. In utility mining, each item has a weight (e.g. unit profit) and can appear more than once in each transaction (e.g. purchase quantity). The utility of an item set represents its importance, which can be measured in terms of weight, profit, cost, quantity or other information depending on the user preference. An itemset is called a high utility itemset (HUI) if its utility is no less than a user-specified minimum utility threshold; otherwise, it is called a low utility itemset. Utility mining is an important task and has a wide range of applications such as website click stream analysis [2], [12], cross-marketing in retail stores mobile commerce environment and biomedical applications.However, efficiently mining HUIs in databases is not an easy task because the downward closure property [1], [8] used in FIM does not hold for the utility of itemsets. In other words, pruning search space for HUI mining is difficult because a superset of a low utility itemset can be high utility. To tackle this problem, the concept of transaction-weighted utilization(TWU) model [12] was introduced to facilitate the performance of the mining task. In this model, an itemset is calledhigh transaction-weighted utilization item-set(HTWUI) if its TWU is no less than min_util, where the TWU of an itemset represents an upper bound on its utility. Therefore, a HUI must be a HTWUI and all the HUIs must be included in the complete set of HTWUIs.To precisely control the output size and discover the itemsets with the highest utilities without setting the thresholds, a promising solution is to redefine the task of mining HUIs as mining top-k high utility itemsets(top-k HUIs). The idea is to let the users specify k, i.e., the number of desired itemsets, instead of specifying the minimum utility thresh-old. Setting k is more intuitive than setting the threshold because k represents the number of itemsets that the users want to find whereas choosing the

threshold depends primarily on database characteristics, which are often unknown to users.

The proposed paper, address all of these challenges by proposing a condensed and meaningful representation of HUIs named closed high utility itemsets(CHUIs), which integrates the concept of closed itemset into high utility itemset mining. Our contributions are four-fold and correspond to resolving the previous four challenges:

- The proposed representation is lossless due to a new structure named utility unit array that allows recovering all HUIs and their utilities efficiently.
- The proposed representation is also compact. Experiments show that it reduces the number of itemsets by several orders of magnitude, especially for datasets containing long high utility itemsets (up to 800 times).
- Proposed three efficient algorithms named AprioriHC(Apriori-based algorithm for mining High utility Closed itemset), AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items) and CHUD(Closed High Utility itemset Discovery) to find this representation. The AprioriHC and AprioriHC-D algorithms employs breadth-first search to find CHUIs and inherits some nice properties from the well-known Apriori algorithm [1]. The CHUD algorithm includes three novel strategies named REG, RML and DCM that greatly enhance its performance. Results show that CHUD is much faster than the state-of-the-art algorithms for mining all HUIs.
- Proposed a top-down method named DAHU (Derive All High Utility itemsets) for efficiently recovering all HUIs from the set of CHUIs. The combination of CHUD and DAHU provides a new way to obtain all HUIs and outperforms UP-Growth, one of the currently best methods for mining HUIs.

II. RELATED WORKS

This section introduces related works about top-k high utility itemset mining, including high utility itemset mining, top-k frequent pattern mining and top-k high utility itemset mining.

2.1 High Utility Itemset Mining

In recent years, high utility itemset mining has received lots of attention and many efficient algorithms have been proposed, such as Two-Phase, IHUP, IIDS, UP-Growth, HUP and HUI-Miner[11]. These algorithms can be generally categorized into two types: two-phase and one-phase algorithms. The main characteristic of two-phase algorithms is that they consist of two phases.

In the first phase, they generate a set of candidates that are potential high utility itemsets. In the second phase, they calculate the exact utility of each candidate found in the first phase to identify high utility itemsets. Two-Phase, IHUP, IIDS and UP-Growth are two-phase based algorithms. UP-Growth is one of the state-of-the-art two-phase algorithms, which incorporates four effective strategies DGU, DGN, DLU and DLN for pruning candidates in the first phase.

2.2 Top-k Pattern Mining

Many studies have been proposed to mine different kinds of top-k patterns, such a stop-k frequent itemsets [3], [16], top-k frequent closed itemsets [3], top-k closed sequential patterns, top-k association rules [6], top-k sequential rules [5], top-k correlation patterns [18], [19], [20] and top-k cosine similarity interesting pairs. What distinguishes each top-k pattern mining algorithm is the type of patterns discovered, as well as the data structures and search strategies that are employed. The choice of data structures and search strategy affect the efficiency of a top-k pattern mining algorithm in terms of both memory and execution time. However, the above algorithms discover top-k patterns according to traditional measures instead of the utility mea-sure. As a consequence, they may miss patterns yielding high utility.

III.CLOSED HIGH UTILITY ITEMSET MINING

In this section, we incorporate the concept of closed itemset with high utility itemset mining to develop a representation named closed high utility itemset. We theoretically prove that this new representation is meaningful, lossless and concise (i.e.,not larger than the set of all HUIs).

3.1 Push Closed Property into High Utility Itemset Mining

The first point that we should discuss is how to incorporate the closed constraint into high utility itemset mining. There are several possibilities. First, we can define the closure on the utility of itemsets. In this case, a high utility itemset is said to be closed if it has no proper superset having the same utility. However, this definition is unlikely to achieve a high reduction of the number of extracted itemsets since not many itemsets have exactly the same utility as their supersets in real datasets. A second possibility is to define the closure on the supports of itemsets. In this case, there are two possible definitions depending on the join order between the closed constraint and the utility constraint.

- Mine all the high utility itemsets first and then apply the closed constraint.
- Mine all the closed itemsets first and then apply the utility constraint.
-

3.2 Efficient Algorithms for Mining Closed High Utility Itemsets

In this section, we introduce three efficient algorithms Aprior-iHC (An Apriori-based algorithm for mining High utility Closed itemsets), AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items) and CHUD (Closed High Utility itemset Discovery) for mining CHUIs. They rely on the TWU-Model[2], [12], [13], [14], [16] and include strategies to improve their performance. All algorithms consist of two phases named Phase I and Phase II. In Phase I, potential closed high utility itemsets(PCHUIs) are found, which are defined as a set of itemsets having an estimated utility (e.g. TWU) no less than abs_min_utility. In Phase II, by scanning the database once, CHUIs are identified from the set of PCHUIs found in Phase I and their utility unit arrays are computed.

The AprioriHC and AprioriHC-D are based on Apriori [1] and the Two-Phase algorithms. They use a horizontal

database and explore the search space of CHUIs in a breadth-first search. The algorithm AprioriHC is regarded as a baseline algorithm in this work and AprioriHC-D is an improved version of AprioriHC. On the other hand, the proposed algorithm CHUD is an extension of Eclat and DCI-Closed algorithms. The CHUD algorithm considers vertical database and mines CHUIs in a depth-first search.

3.2.1 The AprioriHC Algorithm

Initially, a variable k is set to 1. The algorithm performs a data-base 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 b proceeds recursively to generate itemsets having a length greater than k.During the k th iteration, the set of k-HTWUIs Lk is used to generate (k+1)-candidates C_{k+1} by using the Apriori-gen function[1]. Then the algorithm computes TWUs of itemsets in$C_{k+1}$by scanning the database D once.

3.2.2 The AprioriHC-D Algorithm

The AprioriHC-D algorithm is an improvement of AprioriHC. It includes two effective strategies to reduce the number of PCHUIs generated in Phase I that are inspired by the UP-Growth and IIDS algorithms [16]. The first strategy is based on the following definition and properties.

ALGORITHM : AprioriHC-D

Input D : the database ;abs_min_utility;
pCHUI : the set of PCHUIs pCHUI
Output : Thee complete set of CHUIs

1. pCHU :=ø
2. L1:=1-HTWUIs in D
3. D1:=DGU-strategy(D,L1)
4. L1:=1-HTWUIs in D1
5. AprioriHC-D_phase-I(D1,pCHUI,abs_min_utility,L1)
6. AprioriHC-D_phase-II(D1,pCHUI,abs_min_utility)

Fig.1. AprioriHC-D algorithm.

PROCEDURE : AprioriHC-D_Phase-I

Input: Dk :the trimmed database ; abs_min_utility;
pCHUI : the set of PCHUIs
Output : the complete set of PCHUIs

1. Lk+1:=Lk
2. For (K=1; Lk+1≠ ø ;K++)
3. { Ck+1:=Apriori-gen(Lk)
4. Lk+1:=EstimatedUtilityOfItemsets)Dk,Ck+1)
5. Lk:=MarkClosed_Itemsets(Lk,Lk+1)
6. pCHUI:=pCHUI U Lk-1
7. Dk+1:=IIDS_strategy(Dk,Lk+1)
8. }

Fig. 2. AprioriHC-D_Phase-I procedure.

3.2.3 The CHUD Algorithm

In this section, we present an efficient depth-first search algorithm named CHUD(Closed High Utility itemset Discovery) to discover CHUIs. CHUD is an extension of DCI-Closed [15], one of the currently best methods to mine closed itemsets. CHUD is adapted for mining CHUIs and include several effective strategies for reducing the number of candi-dates generated in Phase I.

Similar to the DCI-Closed algorithm, CHUD adopts an Itemset-Tidset pair Tree(IT-Tree) to find CHUIs. In an IT-Tree, each node N(X) consists of an itemsetX, its Tidset g(X), and two ordered sets of items named PREV-SET(X) and POST-SET(X). The IT-Tree is recursively explored by the CHUD algorithm until all closed itemsets that are HTWUIs are generated. Different from the DCI-Closed algorithm, each node N(X) of the IT-Tree is attached with an estimated utility value EstU(X).

PROCEDURE : AprioriHC-D_Phase-II

Input: D1: the database containing no unpromising items;
pCHUI:set if PCHUIs;abs_min_utility
Output: the complete set of CHUIs

1. For (K=1; Lk≠ ø ;K++)
2. {Lk :=K-itemsets in pCHUI
3. For all Xin in Lk do
4. { calculate au(X) and utility unit array of X from L
5. If au(X)≥abs_min_utility then{Output X}
6. }
7. Dk+1 :=IIDS_Strategy(Dk,Lk)
8. }

Fig. 3. AprioriHC-D_Phase-II procedure.

PROCEDURE: calculateESTUtility
Input: g(x): the Tidset of X;TU: a TU table
Output: EstU: the estimated utility of X

1. EstU:=0;
2. For eachTID R€g(X) do
3. { EstU:=EstU + TU.get(R) }
4. Return EstU

Fig. 4. Calculate Est Utility procedure.

A data structure calledtransaction utility table(TU-Table) is adopted for storing the transaction utilities of transac-tions. It is a list of pairs where the first value is a TID Rand the second value is the transaction utility of TR. Given a TID R, the value TU(T_R) can be efficiently retrieved from the TU-Table. Given a node N(X) with its Tidsetg(X) and a TU-Table TU, the estimated utility of the itemset X can be efficiently calculated by the procedure shown in Fig. 4. The main procedure of CHUD is named Main and is shown in Fig. 5. It takes as parameter a database D and the abs_min_utility threshold. CHUD first scans D once to convert D into a vertical database. At the same time, CHUD computes the transaction utility for each transaction TR and calculates TWU of items. When a transaction is retrieved, its Tid and transaction utility are loaded into a global TU-Table named GTU. As previously defined, an item is called a promising item if its estimated utility (e.g. its TWU) is no less than abs_min_utility. After the database scan, promising items are collected into an ordered list O, sorted according to a fixed order such as increasing order of support. Only promising items are kept in O since supersets of unpromising items are not CHUIs (by Property 10). According, the utilities of unpromising items can be removed from the GTU table.This step is performed at line 2 of the Main procedure. Then, CHUD generates candidates in a recursive manner, starting from candidates containing a single promising item and recursively joining items to them to form larger candidates.

ALGORITHM: CHUD

Input: D:the database ;abs_min_utility
Output : Thee complete set of CHUIs

1. InitialDatabaseScan(D)
2. RemoveUtilityUnpromisingItems(0,GTU)
3. For each item ak € 0 do
4. { create node N({ak})
5. CHUD_phase-I(N({ak}),GTU,abs_min_utility)
6. REG_strategy(g(ak)),GTU)}
7. CHUD_phaes-II)D,abs_min_utility)

Fig. 5. CHUD algorithm.

IV.EXPERIMENTAL EVALUATION

In this section, evaluate the performance of the proposed algorithms and compare them with two state-of-the-art algorithms UP-Growth [21] and Two-Phase [17]. Although our methods produce different results from those algorithms, they also consist of two phases. In Phase I, the pro-posed algorithms generate candidates for CHUIs, whereas UP-Growth and Two-Phase generate candidate for HUIs. In Phase II, the proposed algorithms and UP-Growth/Two-Phase respectively identify CHUIs and HUIs from candidates produced in their Phase I. Furthermore, we have also considered the performance of CHUD with DAHU, denoted as CHUD DAHU. CHUD DAHU first applies CHUD to find all CHUIs and then uses DAHU to derive all HUIs from the set of CHUIs generated by CHUD.

The process of CHUD DAHU in Phase I is the same as that of CHUD. In Phase II, CHUD DAHU first identifies CHUIs from candidates and then uses CHUIs to derive all HUIs. In the experiments, we do not combine AprioriHC/AprioriHC-D with DAHU because CHUD outperforms these algorithms, as it will be shown, and they produce the same output. Experiments were performed on a desktop computer with an Intel Core 2 Quad Core Processor @ 2.66 GHz running Windows XP and 2 GB of RAM. All the algorithms were implemented in Java.

Parameter for Synthetic Datasets

Parameter Descriptions	**Default**
D: Total number of transactions	200k
T: Average transaction length	12
N: Number of distinct items	1,000
I: Average size of maximal potentialFIs	10
Q: Maximum number of purchased items in transactions	5

Table 1: Parameter for Synthetic Datasets

Both synthetic and real datasets were used to evaluate the performance of the algorithms. A synthetic dataset T12-I10-N1K-Q5-D200K was generated by the IBM data generator [1].

Characteristics of Datasets

Dataset	**N**	**T**
Mushroom	119	23
Foodmart	1,599	4.4
BmsWebView1	497	2.51
T12-I10-N1K-Q5-D200K	1,000	12

Table 2 shows the characteristics of the above datasets.

Depending on the applications, the characteristics and count distributions of the datasets can be very different. However, there are three kinds of datasets that are commonly encountered in real-life scenarios: (1) dense dataset, (2) sparse dataset, and (3) dataset containing long transactions. In the experiments, we use three real-life datasets Mushroom, Foodmart, BMSWebView1 to respectively rep-resent the above three real cases.

4.1 Experiments on Mushroom Dataset

The performance of the algorithms on the Mushroom data-set is shown in Fig. 6. In Fig. 6a, the execution time of Two-Phase and AprioriHC is similar in Phase I. The reason is that Two-Phase and AprioriHC simply apply the TWU-Model without using effective strategies to reduce the esti-mated utility of candidates in Phase I. Besides, AprioriHC-D runs faster than Two-Phase and AprioriHC. Table 2 shows the number of candidates generated by Two-Phase, AprioriHC and AprioriHC-D in Phase I.

In Fig. 6b, AprioriHC runs faster than Two-Phase because Two-Phase needs to verify the utility of more candidates in Phase II. Though AprioriHC needs to calculate the utility unit array of candidates and Two-Phase does not, this cost is not expensive.

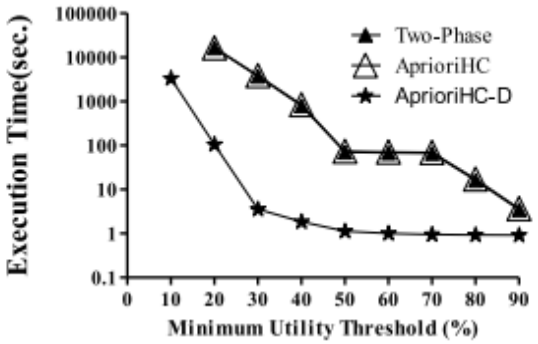

Fig 6a. Execution time for phase I

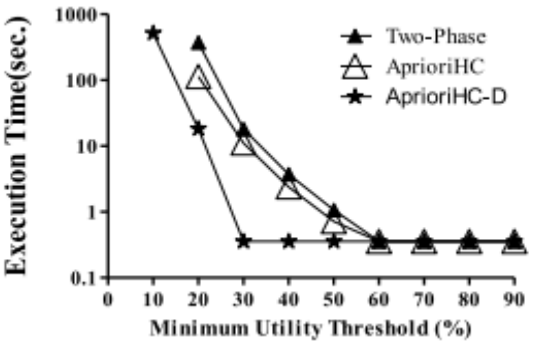

Fig 6b. Execution time for phase II

4.2 Scalability of the Proposed Methods

In this section, evaluated the scalability of the algorithms. The scalability of the proposed algorithms under varied size of potential maximal frequent patterns.

4.3 Memory Usage

It Measure the maximum memory usage of UP-Growth and CHUD in phase I by using the Java API. In general, CHUD uses as much or more memory than UP-Growth because the latter uses a compact trie-based data structure for representing the database that is more memory efficient than a vertical database.

V. CONCLUSION

In this paper, addressed the problem of redundancy in high utility itemset mining by proposing a lossless and compact representation named closed high utility itemsets, which has not been explored so far. To mine this representation, we proposed three efficient algorithms named AprioriHC (Apriori-based approach for mining High utility Closed itemset), AprioriHC-D(AprioriHC algorithm with Discarding unpromising and isolated items) and CHUID(Closed High Utility itemset Discovery). AprioriHC-D is an improved version of AprioriHC, which incorporates strategies DGU and IIDS for pruning candidates. AprioriHC and AprioriHC-D perform a breadth-first search for mining closed high utility itemsets from horizontal database, while CHUD per-forms a depth-first search for mining closed high utility itemsets from vertical database. The strategies incorporated in CHUD are efficient and novel. They have never been used for vertical mining of high utility itemsets and closed high utility itemsets. To efficiently recover all high utility itemsets from closed high utility itemsets, we proposed an efficient method named DAHU (Derive All High Utility item-sets). Results on both real and synthetic datasets show that the proposed representation achieves a massive reduction in the number of high utility itemsets on all real datasets. The combination of CHUD and DAHU is also faster than UP-Growth when DAHU could be applied.

REFERENCE

[1]. R. Agrawal and R. Srikant, "Fast algorithms for mining associa-tion rules," inProc. Int. Conf. Very Large Data Bases, 1994, pp. 487–499.

[2]. C. Ahmed, S. Tanbeer, B. Jeong, and Y. Lee, "Efficient tree struc-tures for high-utility pattern mining in incremental databases,"IEEE Trans. Knowl. Data Eng., vol. 21, no. 12, pp. 1708–1721, Dec.2009.

[3]. K. Chuang, J. Huang, and M. Chen, "Mining top-kfrequent pat-terns in the presence of the memory constraint," VLDB J., vol. 17,pp. 1321–1344, 2008.

[4]. R. Chan, Q. Yang, and Y. Shen, "Mining high-utility itemsets," in Proc. IEEE Int. Conf. Data Mining, 2003, pp. 19–26.

[5]. P. Fournier-Viger and V. S. Tseng, "Mining top-k sequential rules," inProc. Int. Conf. Adv. Data Mining Appl., 2011, pp. 180–194.

[6]. P. Fournier-Viger, C.Wu, andV. S. Tseng, "Mining top-k association rules," in Proc. Int. Conf. Can. Conf. Adv. Artif. Intell., 2012, pp. 61–73.

[7]. P. Fournier-Viger, C. Wu, and V. S. Tseng, "Novel concise repre-sentations of high utility itemsets using generator patterns," in Proc. Int. Conf. Adv. Data Mining Appl. Lecture Notes Comput. Sci., 2014, vol. 8933, pp. 30–43.

[8]. J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without can-didate generation," inProc. ACM SIGMOD Int. Conf. Manag. Data, 2000, pp. 1–12.

[9]. J. Han, J. Wang, Y. Lu, and P. Tzvetkov, "Mining top-k frequent closed patterns without minimum support," in Proc. IEEE Int. Conf. Data Mining, 2002, pp. 211–218.

[10]. S. Krishnamoorthy, "Pruning strategies for mining high utility itemsets,"Expert Syst. Appl., vol. 42, no. 5, pp. 2371–2381, 2015.

[11]. M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," in Proc. ACM Int. Conf. Inf. Knowl. Manag., 2012, pp. 55–64.

[12]. Y. Liu, W. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," in Proc. Utility-Based Data Mining Workshop, 2005, pp. 90–99.

[13]. M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," in Proc. ACM Int. Conf. Inf. Knowl. Manag., 2012, pp. 55–64.

[14]. Y. Li, J. Yeh, and C. Chang, "Isolated items discarding strategy for discovering high-utility itemsets," Data Knowl. Eng., vol. 64, no. 1, pp. 198–217, 2008.

[15]. J. Pisharath, Y. Liu, B. Ozisikyilmaz, R. Narayanan, W. K. Liao, A. Choudhary, and G. Memik, NU-MineBench version 2.0 dataset and technical report [Online]. Available: http://cucis.ece.northwestern.edu/projects/DMS/MineBench.html, 2005.

[16]. G. Pyun and U. Yun, "Mining top-k frequent patterns with combination reducing techniques," Appl. Intell., vol. 41, no. 1, pp. 76–98,2014

[17]. B. Shie, H. Hsiao, V. S. Tseng, and P. S. Yu, "Mining high utility mobile sequential patterns in mobile commerce environments," in Proc. Int. Conf. Database Syst. Adv. Appl. Lecture Notes Comput. Sci., 2011, vol. 6587, pp. 224–238.

[18]. H. Xiong, M. Brodie, and S. Ma, "TOP-COP: Mining TOP-K strongly correlated pairs in large databases," in Proc. IEEE Int. Conf. Data Mining, 2006, pp. 1162–1166.

[19]. H. Xiong, P. Tan, and V. Kumar, "Mining strong affinity association patterns in data sets with skewed support distribution," in Proc. IEEE Int. Conf. Data Mining, 2003, pp. 387–394.

[20]. H. Xiong, P. Tan, and V. Kumar, "Hyperclique pattern discovery," Data Mining Knowl. Discovery, vol. 13, no. 2, pp. 219–242, 2006.

[21]. P. Tzvetkov, X. Yan, and J. Han, "TSP: Mining top-k closed sequential patterns," Knowl. Inf. Syst., vol. 7, no. 4, pp. 438–457, 2005.