11

Mining Multiple-level Association Rules Based on Pre-large Concepts

Tzung-Pei Hong[1, 2], Tzu-Jung Huang[1] and Chao-Sheng Chang[3]
[1]*National University of Kaohsiung*
[2]*National Sun Yat-Sen University*
[3]*I-Shou University*
Kaohsiung, Taiwan, R.O.C.

1. Introduction

The goal of data mining is to discover important associations among items such that the presence of some items in a transaction will imply the presence of some other items. To achieve this purpose, Agrawal and his co-workers proposed several mining algorithms based on the concept of large itemsets to find association rules in transaction data (Agrawal et al., 1993a) (Agrawal et al., 1993b) (Agrawal & Srikant, 1994) (Agrawal & Srikant, 1995). They divided the mining process into two phases. In the first phase, frequent (large) itemsets are found based on the counts by scanning the transaction data. In the second phase, association rules were induced from the large itemsets found in the first phase. After that, several other approaches were also proposed (Fukuda et al., 1996) (Han & Fu, 1995) (Mannila et al., 1994) (Park et al., 1997) (Srikant & Agrawal, 1996) (Han et al., 2000) (Li et al., 2003).

Many of the above algorithms for mining association rules from transactions were executed in level-wise processes. That is, itemsets containing single items were processed first, then itemsets with two items were processed, then the process was repeated, continuously adding one more item each time, until some criteria were met. These algorithms usually considered the database size static and focused on batch mining. In real-world applications, however, new records are usually inserted into databases, and designing a mining algorithm that can maintain association rules as a database grows is thus critically important.

When new records are added to databases, the original association rules may become invalid, or new implicitly valid rules may appear in the resulting updated databases (Cheung et al., 1996) (Cheung et al., 1997) (Lin & Lee, 1998) (Sarda & Srinivas, 1998) (Zhang, 1999). In these situations, conventional batch-mining algorithms must re-process the entire updated databases to find final association rules. Cheung and his co-workers thus proposed an incremental mining algorithm, called FUP (Fast UPdate algorithm) (Cheung et al., 1996), for incrementally maintaining mined association rules and avoiding the shortcomings mentioned above. The FUP algorithm modified the Apriori mining algorithm (Agrawal & Srikant, 1994) and adopted the pruning techniques used in the DHP (Direct Hashing and Pruning) algorithm (Park et al., 1997). It first calculated large itemsets mainly from newly

Source: Data Mining and Knowledge Discovery in Real Life Applications, Book edited by: Julio Ponce and Adem Karahoca,
ISBN 978-3-902613-53-0, pp. 438, February 2009, I-Tech, Vienna, Austria

inserted transactions, and compared them with the previous large itemsets from the original database. According to the comparison results, FUP determined whether re-scanning the original database was needed, thus saving some time in maintaining the association rules. Although the FUP algorithm can indeed improve mining performance for incrementally growing databases, original databases still need to be scanned when necessary. Hong et al. thus proposed a new mining algorithm based on two support thresholds to further reduce the need for rescanning original databases (Hong et al., 2001). They also used a data structure to further improve the performance (Hong et al., 2008).

Most algorithms for association rule mining focused on finding association rules on the single-concept level. However, mining multiple-concept-level rules may lead to discovery of more specific and important knowledge from data. Relevant data item taxonomies are usually predefined in real-world applications and can be represented using hierarchy trees. This chapter thus proposes an incremental mining algorithm to efficiently and effectively maintain the knowledge with a taxonomy based on the pre-large concept and to further reduce the need for rescanning original databases. Since rescanning the database spends much computation time, the mining cost can thus be reduced in the proposed algorithm.

The remainder of this chapter is organized as follows. Some related researches for incremental mining are described in Section 2. Data mining at multiple-level taxonomy is introduced in Section 3. The proposed incremental mining algorithm for multiple-level association rules is described in Section 4. An example to illustrate the proposed algorithm is given in Section 5. Conclusions are summarized in Section 6.

2. Some related researches for incremental mining

In real-world applications, transaction databases grow over time and the association rules mined from them must be re-evaluated because new association rules may be generated and old association rules may become invalid when the new entire databases are considered. Designing efficient maintenance algorithms is thus important.

In 1996, Cheung proposed a new incremental mining algorithm, called FUP (Fast Update algorithm) (Cheung et al., 1996) (Cheung et al., 1997) for solving the above problem. Using FUP, large itemsets with their counts in preceding runs are recorded for later use in maintenance. Assume there exist an original database and newly inserted transactions. FUP divides the mining process into the following four cases (Figure 1):

Case 1: An itemset is large in the original database and in the newly inserted transactions.

Case 2: An itemset is large in the original database, but is not large (small) in the newly inserted transactions.

Case 3: An itemset is not large in the original database, but is large in the newly inserted transactions.

Case 4: An itemset is not large in the original database and in the newly inserted transactions.

Since itemsets in Case 1 are large in both the original database and the new transactions, they will still be large after the weighted average of the counts. Similarly, itemsets in Case 4 will still be small after the new transactions are inserted. Thus Cases 1 and 4 will not affect the final association rules. Case 2 may remove existing association rules, and case 3 may add new association rules. FUP thus processes these four cases in the manner shown in Table 1.

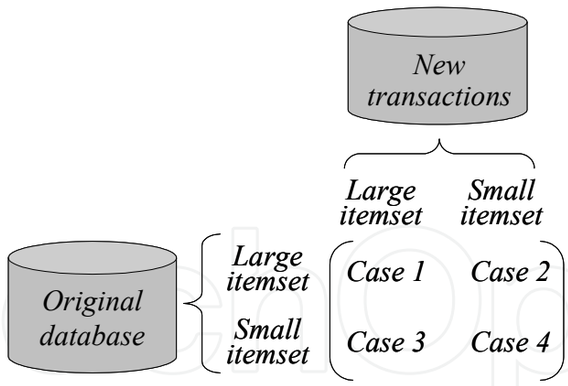

Fig. 1. Four cases in the FUP algorithm

Cases: Original – New	Results
Case 1: Large – Large	Always large
Case 2: Large – Small	Determined from existing information
Case 3: Small – Large	Determined by rescanning original database
Case 4: Small – Small	Always small

Table 1. Four cases and their FUP results

FUP thus focuses on the newly inserted transactions and can save some processing time in rule maintenance. But FUP still has to scan an original database for managing Case 3 in which a candidate itemset is large in newly inserted transactions but is small in the original database. This situation may often occur when the number of newly inserted transactions is small. For example, suppose only one transaction is inserted into a database. In this situation, each itemset in the transaction is large. Case 3 thus needs to be processed in a more efficient way.

Hong et al. thus propose a new mining algorithm based on pre-large itemsets to further reduce the need for rescanning original databases (Hong et al., 2001). A pre-large itemset is not truly large, but promises to be large in the future. A lower support threshold and an upper support threshold are used to realize this concept. The upper support threshold is the same as that used in the conventional mining algorithms. The support ratio of an itemset must be larger than the upper support threshold in order to be considered large. On the other hand, the lower support threshold defines the lowest support ratio for an itemset to be treated as pre-large. An itemset with its support ratio below the lower threshold is thought of as a small itemset. Pre-large itemsets act like buffers in the incremental mining process and are used to reduce the movements of itemsets directly from large to small and vice-versa.

Considering an original database and transactions newly inserted using the two support thresholds, itemsets may thus fall into one of the following nine cases illustrated in Figure 2. Cases 1, 5, 6, 8 and 9 above will not affect the final association rules according to the weighted average of the counts. Cases 2 and 3 may remove existing association rules, and cases 4 and 7 may add new association rules. If we retain all large and pre-large itemsets with their counts after each pass, then cases 2, 3 and case 4 can be handled easily. Also, in the maintenance phase, the ratio of new transactions to old transactions is usually very small. This is more apparent when the database is growing larger. An itemset in case 7 cannot possibly be large for the entire updated database as long as the number of

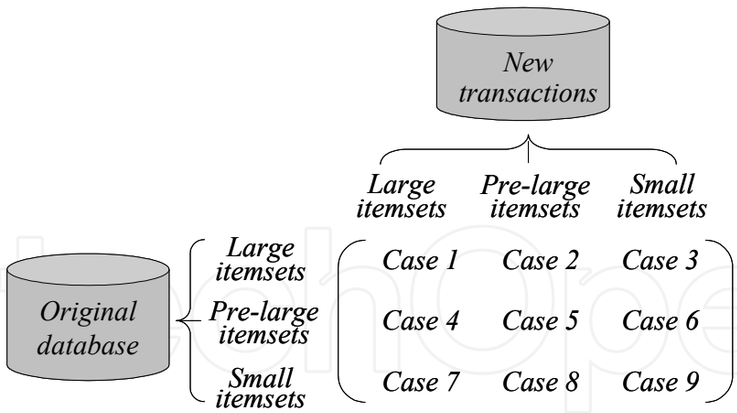

Fig. 2. Nine cases arising from adding new transactions to existing database

transactions is small when compared to the number of transactions in the original database. Let S_l and S_u be respectively the lower and the upper support thresholds, and let d and t be respectively the numbers of the original and new transactions. They showed that an itemset that is small (neither large nor pre-large) in the original database but is large in newly inserted transactions is not large for the entire updated database if the following condition is satisfied:

$$t \leq \frac{(S_u - S_l)d}{1 - S_u}. \tag{1}$$

In this chapter, we will generalize Hong et al's approach to maintain the association rules with taxonomy.

4. Mining multiple-level association rules

Most algorithms for association rule mining focused on finding association rules on the single-concept level. However, mining multiple-concept-level rules may lead to discovery of more specific and important knowledge from data. Relevant data item taxonomies are usually predefined in real-world applications and can be represented using hierarchy trees. Terminal nodes on the trees represent actual items appearing in transactions; internal nodes represent classes or concepts formed by lower-level nodes. A simple example is given in Figure 3.

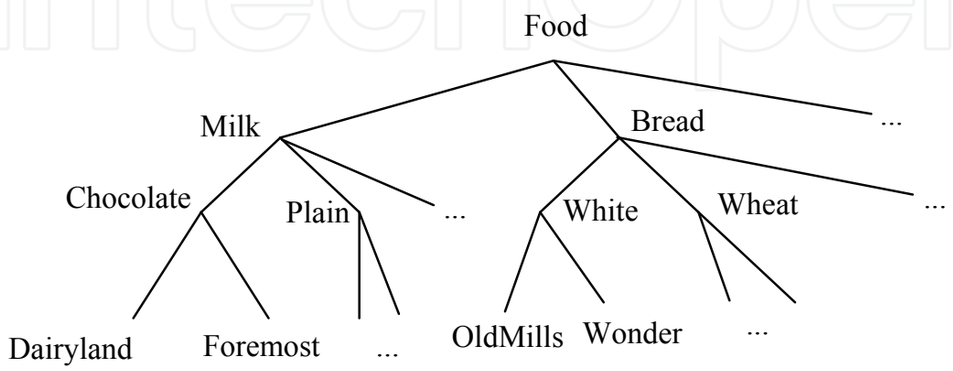

Fig. 3. An example of taxonomy

In Figure 3, the root node is at level 0, the internal nodes representing categories (such as "milk") are at level 1, the internal nodes representing flavors (such as "chocolate") are at level 2, and the terminal nodes representing brands (such as "Foremost") are at level 3. Only terminal nodes appear in transactions.

Han and Fu proposed a method for finding level-crossing association rules at multiple levels (Han & Fu, 1995). Nodes in predefined taxonomies are first encoded using sequences of numbers and the symbol "*" according to their positions in the hierarchy tree. For example, the internal node "Milk" in Figure 3 would be represented by 1**, the internal node "Chocolate" by 11*, and the terminal node "Dairyland" by 111. A top-down progressively deepening search approach is used and exploration of "level-crossing" association relationships is allowed. Candidate itemsets on certain levels may thus contain other-level items. For example, candidate 2-itemsets on level 2 are not limited to containing only pairs of large items on level 2. Instead, large items on level 2 may be paired with large items on level 1 to form candidate 2-itemsets on level 2 (such as {11*, 2**}).

5. The proposed incremental mining algorithm for multiple-level association rules

The proposed incremental mining algorithm integrates Hong et al's pre-large concepts and Han and Fu's multi-level mining method. Assume d is the number of transactions in the original database. A variable, c, is used to record the number of new transactions since the last re-scan of the original database. Details of the proposed mining algorithm are given below.

The incremental multi-level mining algorithm:

INPUT: A set of large itemsets and pre-large itemsets in the original database consisting of $(d+c)$ transactions, a set of t new transactions, a predefined taxonomy, a lower support threshold S_l, an upper support threshold S_u, and a predefined confidence value λ.

OUTPUT: A set of multi-level association rules for the updated database.

STEP 1: Calculate the safety number f of new transactions as follows:

$$f = \left\lfloor \frac{(S_u - S_l)d}{1 - S_u} \right\rfloor. \tag{2}$$

STEP 2: Set l = 1, where l records the level of items in taxonomy.

STEP 3: Set k = 1, where k records the number of items in itemsets.

STEP 4: Find all the candidate k-itemsets C_k and their counts in the new transactions.

STEP 5: Divide the candidate k-itemsets into three parts according to whether they are large, pre-large or small in the original database.

STEP 6: For each itemset I in the originally large k-itemsets L_k^D, do the following substeps:

Substep 6-1: Set the new count $S^U(I) = S^T(I) + S^D(I)$.

Substep 6-2: If $S^U(I)/(d+t+c) \geq S_u$, then assign I as a large itemset, set $S^D(I) = S^U(I)$ and keep I with $S^D(I)$;
otherwise, if $S^U(I)/(d+t+c) \geq S_l$, then assign I as a pre-large itemset, set $S^D(I) = S^U(I)$ and keep I with $S^D(I)$;
otherwise, neglect I.

STEP 7: For each itemset I in the originally pre-large itemset P_k^D, do the following substeps:

Substep 7-1: Set the new count $S^U(I) = S^T(I) + S^D(I)$.

Substep 7-2: If $S^U(I)/(d+t+c) \geq S_u$, then assign I as a large itemset, set $S^D(I) = S^U(I)$ and keep I with $S^D(I)$;

otherwise, if $S^U(I)/(d+t+c) \geq S_l$, then assign I as a pre-large itemset, set $S^D(I) = S^U(I)$ and keep I with $S^D(I)$;

otherwise, neglect I.

STEP 8: For each itemset I in the candidate itemsets that is not in the originally large itemsets L_k^D or pre-large itemsets P_k^D, do the following substeps:

Substep 8-1: If I is in the large itemsets L_k^T or pre-large itemsets P_k^T from the new transactions, then put it in the rescan-set R, which is used when rescanning in Step 9 is necessary.

Substep 8-2: If I is small for the new transactions, then do nothing.

STEP 9: If $t + c \leq f$ or R is null, then do nothing; otherwise, rescan the original database to determine whether the itemsets in the rescan-set R are large or pre-large.

STEP 10: Form candidate $(k+1)$-itemsets C_{k+1} from finally large and pre-large k-itemsets ($L_k^U \cup P_k^U$) that appear in the new transactions.

STEP 11: Set $k = k+1$.

STEP 12: Repeat STEPs 5 to 11 until no new large or pre-large itemsets are found.

STEP 13: Prune the kept large or pre-large itemsets in the next level which are not the descendents of those found after STEP 12.

STEP 14: Set $l = l + 1$.

STEP 15: Repeat STEPs 3 to 14 until all levels are processed or there are no large and pre-large itemsets on level $l - 1$.

STEP 16: Modify the association rules according to the modified large itemsets.

STEP 17: If $t + c > f$, then set $d=d+t+c$ and set $c=0$; otherwise, set $c=t+c$.

After Step 17, the final multi-level association rules for the updated database have been determined.

6. An example

An example is given to illustrate the proposed mining algorithm. Assume the original database includes 8 transactions as shown in Table 2.

Each transaction includes a transaction ID and some purchased items. For example, the eighth transaction consists of three items: Foremost plain milk, Old Mills white bread, and Wonder wheat bread. Assume the predefined taxonomy is as shown in Figure 4.

The food in Figure 4 falls into four main classes: milk, bread, cookie and beverage. Milk can further be classified into chocolate milk and plain milk. There are two brands of chocolate milk, Dairyland and Foremost. The other nodes can be similarly explained. Each item name in the taxonomy can then be encoded by Han and Fu's approach. Results are shown in Table 3.

TID	ITEMS
100	Dairyland chocolate milk, Foremost chocolate milk, Old Mills white bread, Wonder white bread, Linton green tea beverage
200	Dairyland chocolate milk, Foremost chocolate milk, Dairyland plain milk, Old Mills wheat bread, Wonder wheat bread, present lemon cookies, 77 lemon cookies
300	Foremost chocolate milk, Old Mills white bread, Old Mills wheat bread, 77 chocolate cookies, 77 lemon cookies
400	Dairyland chocolate milk, Old Mills white bread, 77 chocolate cookies
500	Old Mills white bread, Wonder wheat bread
600	Dairyland chocolate milk, Foremost plain milk, Wonder white bread, Nestle black tea beverage
700	Dairyland chocolate milk, Foremost chocolate milk Dairyland plain milk, Old Mills white bread
800	Foremost plain milk, Old Mills white bread, Wonder wheat bread

Table 2. The original database in this example

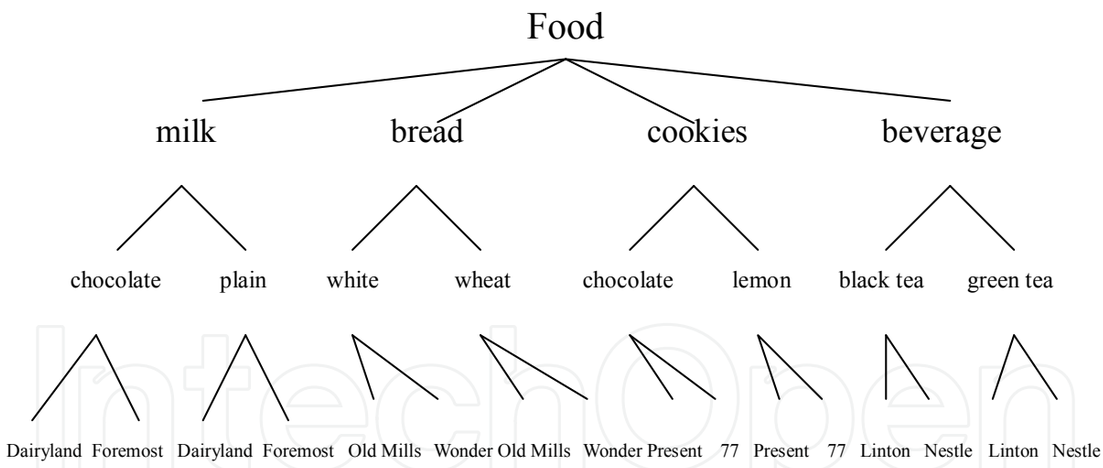

Fig. 4. The predefined taxonomy in this example

For example, the item "Foremost chocolate milk" is encoded as '112', in which the first digit '1' represents the code 'milk' on level 1, the second digit '1' represents the flavor 'chocolate' on level 2, and the third digit '2' represents the brand 'Foremost' on level 3. All the transactions shown in Table 2 are then encoded using the above coding table. Results are shown in Table 4

For S_l=30% and S_u=50%, the sets of large itemsets and pre-large itemsets on any level for the given original transaction database are shown in Table 5 and 6, respectively. They are then kept for later incremental mining.

Item name (terminal node)	Code	Item name (internal node)	Code
Dairyland chocolate milk	111	milk	1**
Foremost chocolate milk	112	bread	2**
Dairyland plain milk	121	cookies	3**
Foremost plain milk	122	beverage	4**
Old Mills white bread	211	chocolate milk	11*
Wonder white bread	212	plain milk	12*
Old Mills wheat bread	221	white bread	21*
Wonder wheat bread	222	wheat bread	22*
Present chocolate cookies	311	chocolate cookies	31*
77 chocolate cookies	312	lemon cookies	32*
Present lemon cookies	321	black tea beverage	41*
77 lemon cookies	322	green tea beverage	42*
Linton black tea beverage	411		
Nestle black tea beverage	412		
Linton green tea beverage	421		
Nestle green tea beverage	422		

Table 3. Codes of item names

TID	ITEMS
100	111, 112, 211, 212, 421
200	111, 112, 121, 221, 222, 322, 321
300	112, 211, 221, 312, 322
400	111, 211, 312
500	211, 222
600	111, 122, 212, 412
700	111, 112, 121, 211
800	122, 211, 222

Table 4. Encoded transaction data in the example

Level-1	Count	Level-2	Count	Level-3	Count
{1**}	7	{11*}	6	{111}	5
{2**}	8	{12*}	4	{112}	4
{1**, 2**}	7	{21*}	7	{211}	6
		{22*}	4		
		{11*, 21*}	5		

Table 5. The large itemsets on all levels for the original database

Level-1	Count	Level-2	Count	Level-3	Count
{3**}	3	{11*, 12*}	3	{222}	3
{1**, 3**}	3	{12*, 21*}	3	{111, 112}	3
{2**, 3**}	3	{21*, 22*}	3	{111, 211}	3
{1**, 2**, 3**}	3			{112, 211}	3

Table 6. The pre-large itemsets on all levels for the original database

Assume now the two new transactions shown in Table 7 are inserted to the original database.

New transactions	
TID	Items
900	112, 221, 311, 412
1000	111, 122, 412, 422

Table 7. Two new transactions

The proposed mining algorithm proceeds as follows. The variable c is initially set at 0.
STEP 1: The safety number f for new transactions is calculated as:

$$f = \left\lfloor \frac{(S_u - S_l)d}{1 - S_u} \right\rfloor = \left\lceil \frac{(0.5 - 0.3)8}{1 - 0.5} \right\rceil = 3. \quad (3)$$

STEP 2: l is set to 1, where l records the level of items in taxonomy.
STEP 3: k is set to 1, where k records the number of items in itemsets currently processed.
STEP 4:All candidate 1-itemsets C_1 on Level 1 and their counts from the two new transactions are found, as shown in Table 8.

Candidate 1-itemsets	
Items	Count
{1**}	2
{2**}	1
{3**}	1
{4**}	2

Table 8. All candidate 1-itemsets on level

STEP 5: From Table 8, all candidate 1-itmesets {1**}{2**}{3**}{4**} are divided into three parts: {1**}{2**}, {3**}, and {4**} according to whether they are large, pre-large or small in the original database. Results are shown in Table 9.

Originally large 1-itemsets		Originally pre-large 1-itemsets		Originally small 1-itemsets	
Items	Count	Items	Count	Items	Count
{1**}	2	{3**}	1	{4**}	2
{2**}	1				

Table 9. Three partitions of all candidate 1-itemsets from the two new transactions

STEP 6: The following substeps are done for each of the originally large 1-itemsets {1**}{2**}:

Substep 6-1: The total counts of the candidate 1-itemsets {1**}{2**} are calculated using $S^T(I)+ S^D(I)$. Table 10 shows the results.

Items	Count
{1**}	9
{2**}	9

Table 10. The total counts of {1**}{2**}

Substep 6-2: The new support ratios of {1**}{2**} are calculated. For example, the new support ratio of {1**} is $9/(8+2+0) \geq 0.5$. {1**} is thus still a large itemset. In this example, both {1**} and {2**} are large. {1**}{2**} with their new counts are then retained in the large 1-itemsets for the entire updated database.

STEP 7: The following substeps are done for itemset {3**}, which is originally pre-large:

Substep 7-1: The total count of the candidate 1-itemset {3**} is calculated using $S^T(I)+ S^D(I)$ (= 4).

Substep 7-2: The new support ratio of {3**} is $4/(8+2+0) \leq 0.5$. {3**} isn't a large 1-itemset for the whole updated database. {3**} with its new count is, however, a pre-large 1-itemset for the entire updated database.

STEP 8: Since the itemset {4**}, which was originally neither large nor pre-large, is large for the new transactions, it is put in the rescan-set R, which is used when rescanning in Step 9 is necessary.

STEP 9: Since $t + c = 2 + 0 \leq f$ (=3), rescanning the database is unnecessary, so nothing is done.

STEP 10: After STEPs 8 and 9, the final large 1-itemsets for the entire updated database are {1**}{2**} and the final pre-large 1-itemset is {3**}. Since all of them are in the new transactions, the candidate 2-itemsets are shown in Table 11.

Candidate 2-itemsets
{1**, 2**}
{1**, 3**}
{2**, 3**}

Table 11. All candidate 2-itemsets for the new transactions

STEP 11: $k = k + 1 = 2$.

STEP 12: STEPs 5 to 11 are repeated to find large and pre-large 2-itemsets on level 1. Results are shown in Table 12.

Large 2-Itemsets		Pre-large 2-Itemsets	
Items	Count	Items	Count
{1**, 2**}	8	{2**, 3**}	4
		{1**, 3**}	4

Table 12. All large and pre-large 2-itemsets on level 1 for the updated database

Large or pre-large 3-itemsets are then found in the same way. The results are shown in Table 13.

Large 3-Itemsets		Pre-large 3-Itemsets	
Items	Count	Items	Count
		{1**, 2**, 3**}	3

Table 13. All large and pre-large 3-itemsets on level 1 for the updated database

STEP 13: The large and pre-large itemsets on level 1 are then used to prune the originally kept itemsets on level 2. If an itemset originally kept on level 2 is not a descendent of any one on level 1, it is pruned. In this example, since all itemsets originally kept on level 2 are descendents of those on level 1, no pruning is made.

STEP 14: $l = l + 1 = 2$. Large and pre-large itemsets on the second level are to be found.

STEP 15: Steps 3 to 14 are repeated to find all large and pre-large itemsets on level 2. The results are shown in Table 14.

Large itemsets			Pre-large itemsets		
1 item	2 items	3 items	1 item	2 items	3 items
{11*}	{11*, 21*}		{31*}	{11*, 12*}	
{12*}				{11*, 22*}	
{21*}				{12*, 21*}	
{22*}				{21*, 22*}	
				{11*, 31*}	

Table 14. All the large and pre-large itemsets on level 2 for the updated database

Similarly, all large and pre-large itemsets on level 3 are found and shown in Table 15.

Large itemsets			Pre-large itemsets		
1 item	2 items	3 items	1 item	2 items	3 items
{111}			{122}	{111, 112}	
{112}			{221}	{111, 211}	
{211}			{222}	{112, 211}	
				{112, 221}	

Table 15. All the large and pre-large itemsets on level 3 for the updated database

The large itemsets on all levels are listed in Table 16.

Level-1	Level-2	Level-3
{1**}	{11*}	{111}
{2**}	{12*}	{112}
{1**, 2**}	{21*}	{211}
	{22*}	
	{11*, 21*}	

Table 16. The large itemsets on all levels for the updated database

STEP 16: Assume the minimum confidence value is set at 0.7. The association rules are then modified according to the modified large itemsets as follows:

1** ⇒ 2** (Confidence=8/9),
2** ⇒ 1** (Confidence=8/9),
21* ⇒ 11* (Confidence=5/7),
11* ⇒ 2** (Confidence=7/8),
2** ⇒ 11* (Confidence=7/9), and
21* ⇒ 1** (Confidence=6/7).

STEP 17: Since t (= 2) + c (= 0) < f (= 3), $c = t + c = 2 + 0 = 2$.

After Step 17, the final association rules for the updated database are found.

7. Conclusion

In this chapter, we have proposed an incremental mining algorithm for multiple-level association rules. It may lead to discovery of more specific and important knowledge from data. The proposed algorithm is based on the pre-large concept and can efficiently and effectively mine knowledge with a taxonomy in an incremental way. The large itemsets play a very critical role to reduce the need for rescanning original databases. Since rescanning the database spends much computation time, the mining cost can thus be greatly reduced in the proposed algorithm. An example is given to illustrate the proposed approach. It can also be easily observed from the example that much rescanning can be avoided. Using the concept of pre-large itemises is thus a good way to incremental mining, no matter for single levels or multiple levels.

8. References

R. Agrawal, T. Imielinksi and A. Swami, "Mining association rules between sets of items in large database", The ACM SIGMOD Conference, pp. 207-216, Washington DC, USA, 1993. (a)

R. Agrawal, T. Imielinksi and A. Swami, "Database mining: a performance perspective", IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 6, pp. 914-925, 1993. (b)

R. Agrawal and R. Srikant, "Fast algorithm for mining association rules," The International Conference on Very Large Data Bases, pp. 487-499, 1994.

R. Agrawal and R. Srikant, "Mining sequential patterns," The Eleventh IEEE International Conference on Data Engineering, pp. 3-14, 1995.

R. Agrawal, R. Srikant and Q. Vu, "Mining association rules with item constraints," The Third International Conference on Knowledge Discovery in Databases and Data Mining, pp. 67-73, Newport Beach, California, 1997.

D.W. Cheung, J. Han, V.T. Ng, and C.Y. Wong, "Maintenance of discovered association rules in large databases: An incremental updating approach," The Twelfth IEEE International Conference on Data Engineering, pp. 106-114, 1996.

D.W. Cheung, V.T. Ng, and B.W. Tam, “Maintenance of discovered knowledge: a case in multi-level association rules,” The Second International Conference on Knowledge Discovery and Data Mining (KDD-96), pp. 307-310, 1996.

D.W. Cheung, S.D. Lee, and B. Kao, “A general incremental technique for maintaining discovered association rules,” In Proceedings of Database Systems for Advanced Applications, pp. 185-194, Melbourne, Australia, 1997.

T. Fukuda, Y. Morimoto, S. Morishita and T. Tokuyama, "Mining optimized association rules for numeric attributes," The ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 182-191, 1996.

J. Han and Y. Fu, “Discovery of multiple-level association rules from large database,” The Twenty-first International Conference on Very Large Data Bases, pp. 420-431, Zurich, Switzerland, 1995.

J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” The 2000 ACM SIGMOD International Conference on Management of Data, pp. 1-12, 2000.

T. P. Hong, C. Y. Wang and Y. H. Tao, “A new incremental data mining algorithm using pre-large itemsets," Intelligent Data Analysis, Vol. 5, No. 2, 2001, pp. 111-129.

T. P. Hong, J. W. Lin, and Y. L. Wu, “Incrementally fast updated frequent pattern trees", Expert Systems with Applications, Vol. 34, No. 4, pp. 2424 – 2435, 2008. (SCI)

Y. Li, P. Ning, X. S. Wang and S. Jajodia. “Discovering calendar-based temporal association rules,” Data & Knowledge Engineering, Vol. 44, No. 2, pp. 193-218, 2003.

M. Y. Lin and S. Y. Lee, “Incremental update on sequential patterns in large databases,” The Tenth IEEE International Conference on Tools with Artificial Intelligence, pp. 24-31, 1998.

H. Mannila, H. Toivonen, and A. I. Verkamo, “Efficient algorithm for discovering association rules,” The AAAI Workshop on Knowledge Discovery in Databases, pp. 181-192, 1994.

J. S. Park, M. S. Chen, P. S. Yu, “Using a hash-based method with transaction trimming for mining association rules,” IEEE Transactions on Knowledge and Data Engineering, Vol. 9, No. 5, pp. 812-825, 1997.

N. L. Sarda and N. V. Srinivas, “An adaptive algorithm for incremental mining of association rules,” The Ninth International Workshop on Database and Expert Systems, pp. 240-245, 1998.

R. Srikant and R. Agrawal, “Mining generalized association rules,” The Twenty-first International Conference on Very Large Data Bases, pp. 407-419, Zurich, Switzerland, 1995.

R. Srikant and R. Agrawal, “Mining quantitative association rules in large relational tables,” The 1996 ACM SIGMOD International Conference on Management of Data, pp. 1-12, Montreal, Canada, 1996.

S. Zhang, "Aggregation and maintenance for database mining," Intelligent Data Analysis, Vol. 3, No. 6, pp. 475-490, 1999.

Data Mining and Knowledge Discovery in Real Life Applications

Edited by Julio Ponce and Adem Karahoca

ISBN 978-3-902613-53-0
Hard cover, 436 pages
Publisher I-Tech Education and Publishing
Published online 01, January, 2009
Published in print edition January, 2009

This book presents four different ways of theoretical and practical advances and applications of data mining in different promising areas like Industrialist, Biological, and Social. Twenty six chapters cover different special topics with proposed novel ideas. Each chapter gives an overview of the subjects and some of the chapters have cases with offered data mining solutions. We hope that this book will be a useful aid in showing a right way for the students, researchers and practitioners in their studies.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Tzung-Pei Hong, Tzu-Jung Huang and Chao-Sheng Chang (2009). Mining Multiple-level Association Rules Based on Pre-large Concepts, Data Mining and Knowledge Discovery in Real Life Applications, Julio Ponce and Adem Karahoca (Ed.), ISBN: 978-3-902613-53-0, InTech, Available from: http://www.intechopen.com/books/data_mining_and_knowledge_discovery_in_real_life_applications/mining_multiple-level_association_rules_based_on_pre-large_concepts

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen