Mining Frequent Closed Itemsets with the Frequent Pattern List

Fan-Chen Tseng, Ching-Chi Hsu*, and Henry Chen

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan, 106
cchsu@csie.ntu.edu.tw

Abstract

*The mining of the complete set of frequent itemsets will lead to a huge number of itemsets. Fortunately, this problem can be reduced to the mining of frequent closed itemsets (FCIs), which results in a much smaller number of itemsets. The approaches to mining frequent closed itemsets can be categorized into two groups: those with candidate generation and those without. In this paper, we propose an approach to mining frequent closed itemsets without candidate generation: with a data structure called the Frequent Pattern List (FPL). We designed the algorithm **FPLCI-Mining** to mine the frequent closed itemsets (FCIs). Experimental result shows that our method is faster than the previously existing ones.*

***Keywords**: frequent closed itemset, frequent pattern list*

**Corresponding author: Fax: 886-2-23628167,*
Tel: 886-2-2391-7406,
Email: cchsu@csie.ntu.edu.tw

1. Introduction and Problem Definitions

The mining of the complete set of frequent patterns often leads to a huge number of results, and the effectiveness of the association rules derived from them will be decreased. Fortunately, Pasquier et al. showed that this problem could be solved by mining only frequent closed itemsets (FCIs), which are a small portion of the complete set of solutions. They developed an *A-Close* algorithm [1], which took the generation-and-test approach, for mining FCIs. Recently, Pei, Han, and Mao designed an algorithm *CLOSET* [2] for mining FCIs without candidate generation by a combination of FP-tree and projected database. Here we use a simpler and more efficient data structure, the *frequent pattern list* (**FPL**) [3], for mining frequent closed itemsets without candidate generation. We redefine the **frequent closed itemset**, and then develop our approach, **FPLCI-Mining**, for mining the frequent closed itemsets.

Let I = {i1, i2, ...in} be a set of items. An itemset X is a non-empty subset of I. A transaction database DB is a set of transactions. Each transaction Tx is a pair <tid, X>, where tid is a unique transaction identifier, and X is an itemset. A transaction Tx = <tid, X> is said to contains an itemset Y, if $Y \subseteq X$. Every item in Y is said to be contained in Tx if Y is contained in Tx. With these descriptions, we have the following definitions:

Definition 1. (Maximal frequent itemset).
A frequent itemset is called a **maximal frequent itemset** if there is no other frequent itemset to be its proper superset.

Definition 2. (Frequent closed itemset.)
A frequent closed itemset is either a maximal frequent itemset, or a frequent itemset whose support is higher than the supports of all its proper supersets.

2. Algorithm for mining frequent closed itemsets with FPL

Based on Definition 2, we take the following strategy for mining frequent closed itemsets (FCIs): finding frequent itemsets with items as many as possible, and with supports as high as possible. We formally define the algorithm FPLCI-Mining of mining frequent closed itemsets with FPL:

Algorithm FPLCI-Mining:

Input: FPL constructed using a transaction database DB and a support threshold *t*.
Output: The complete set of frequent **closed** itemsets.
Method: Call **FPLCI-Mining**(*FPL,n, t, ? , ?*)
FPLCI-Mining (*FPL, n, min_sup, parent_itemset, FCIS*)
{**For** *j* = *n* **to** *1*
{ // check the necessity to visit item node *j*
If (there is no existing FCI in *FCIS* that contains {*item j* }U *parent_itemset* and has a support equal to (*item node*

0-7695-1119-8/01 $17.00 © 2001 IEEE

j).*count*) **Then** { // bit counting1. Examine the *bit-count array* of item node *j*, and ignore the LSB, which corresponds to item *j*, since it will always be included in the solution;2. Divide the surviving bits (whose counts are above *min_sup*) into two groups:
Group One: bit counts equal to (*item node j*).*count*.
Group Two: bit counts less than (*item node j*).*count*.;
3. Generate a FCI, denoted as *fci_GroupOne*: *fci_GroupOne* = {*Group one* items}U{*item j*}U *parent_itemset*, with count = (*item node j*).*count*;
4. Use all the signatures in node item *j*, with the surviving bits in *Group Two* as filtering mask to keep their corresponding items, to form *fci_GroupOne*'s *conditional* database and construct a *conditional* FPL $FPL_{fci\_GroupOne}$ from this database; Let $FPL_{fci\_GroupOne}$ have *m* item nodes;
5. Call **FPLCI-Mining** ($FPL_{fci\_GroupOne}$, *m*, *min_sup*, *fci_GroupOne*, *FCIS*);
} // end of bit counting
// conducting signature trimming and migration
For each transaction *Tx* in item node *j*, consider its full-length *j*–bit signature
{1. Trim the LSB (corresponding to item *j*) and then trim all the trailing 0-bits;
2. Find the least significant 1-bit and find *the item* corresponding to this bit;
3. Migrate the trimmed signature to the item node containing *this item*;
4. For the *bit-count array* of the target item node, increment the count values by one for the elements that correspond to the 1-bits in *Tx*.}
Remove item node *j* from the FPL;
// end of signature trimming and migration
} // end of for loop of index *j*
} // end of procedure FPLCI-Mining

3. Experimental Results and Discussion

We test our algorithms on the synthetic data set T25.I20.D100K. There are 10K items. The number of transactions is set to 100K. The average transaction size and average maximal potentially frequent itemset size are set to 25 and 20, respectively. To compare our method with the existing ones: A-Close [1], CHARM [4], and CLOSET [2], we run our program on a Pentium 233-MHz PC with 128 megabytes main memory, running Microsoft Windows 98. The algorithms are implemented with Microsoft Visual C++ 6.0. The run time is the total execution time, including the time for disk I/O and the time for constructing the FPL from the original databases. The result is shown in Figure 1. From the result we see that FPLCI-Mining is much faster than A-Close and CHARM, and is also faster than CLOSET by 12 percent in average. The efficiency of our method is due to the following factors: no candidate generation and testing, simple data structures, and simple operations. Besides, the optimization techniques of CLOSET can be implemented in our method. For optimization 1, we encode the database into the global FPL, and the conditional FPL can be derived directly in step 4 of FPLCI-Mining. Optimization 2 is implemented in step 2 and 3 of FPLCI-Mining to extract items appearing in every transaction of the item node; that is, the Group-One items. The single-path FP-tree of optimization 3 can also be detected in our FPL data structure, and the same technique can be used to speed up the mining process. Finally, for optimization 4, the checking before bit counting for the necessity to visit the item node prunes the search space.

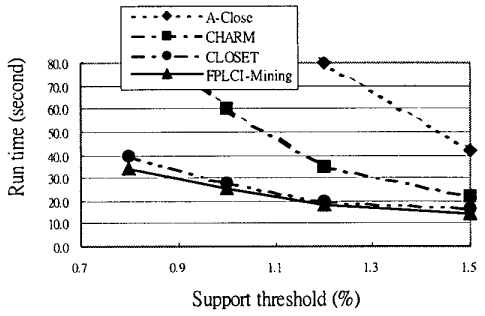

Figure 1. Experimental results

4 Conclusions

In this paper we proposed an efficient approach, FPLCI-Mining, to mining frequent closed itemsets with the simple structure FPL (frequent pattern list). There are several issues related to FPL-based mining. For example, the patterns in the transaction signatures should be studied to derive more efficient algorithms.

References

[1] Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L., "Discovering frequent closed itemsets for association rules." Proc. 7th Int. Conf. Database Theory (ICDT'99), pp. 398-416, Jan. 1999

[2] Pei, J., Han, J., Mao, R., CLOSET, "An efficient algorithm for mining frequent closed itemsets." Proc. 2000 ACMSIGMOD Int. Workshop Data Mining and Knowledge Discovery (DMKD00), pp. 11-20, May 2000

[3] Tseng, Fan-Chen, Hsu, Ching-Chi, "Generating Frequent Patterns with the Frequent Pattern List." Proc. The Fifth Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 376-386, April 16-18, 2001. Hong Kong.

[4] Zaki, M. et al, "An Efficient Algorithm for Closed Association Rule Mining." Technical Report 99-10, Computer Science, Rensselaer Polytechnic Institute, 1999