International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Efficient Analysis of Closed Frequent Pattern Set Mining Approach

Javeriya Naaz I. Syed[1], Rajeshri R. Shelke[2]

[1, 2]Computer Science and Engineering, Shree HVPM's COET, Amravati, Maharashtra, India

Abstract: ***Many researchers' unreal the ideas to get the frequent item sets/ pattern sets. Some frequent pattern mining often produces a large number of frequent patterns, which imposes a great challenge on visualizing, understanding and further analysis of the generated patterns. This emerges the need for finding small number frequent occurring patterns. The time needed for generating frequent pattern sets plays associate important role. Some algorithms are designed, considering solely the time issue. Our study includes analysis of algorithms that are used to find frequent pattern sets by using the technique of closed pattern sets.***

Keywords: frequent pattern, closed pattern, analysis, frequent items, pattern mining

1. Introduction

Data mining is the process of finding new patterns from large amount of data. Data mining is a process of extraction of useful information and patterns from huge data. It is also called as knowledge discovery process, knowledge mining from data, knowledge extraction or data/ pattern analysis. The goal of this technique is to find patterns that were previously unknown. Once these patterns are found they can further be used to make certain decisions for development of their businesses. Data mining aims to discover implicit, previously unknown,and potentially useful information that is embedded in data. Asa common data mining task, frequent itemset mining [1–3] looksfor itemsets (i.e., sets of items) that are frequently co-occurringtogether.It has been well recognized that frequent pattern mining plays an essential role in many important data mining tasks, e.g. associations, sequential patterns , episodes, partial periodicity, etc. However, it is also well known that frequent pattern mining often generates a very large number of frequent itemsets and rules, which reduces not only efficiency but also effectiveness of mining since users have to sift through a large number of mined rules. There is an interesting alternative, proposed recently by Pasquier et al. [4]: instead of mining the complete set of frequent itemsets and their associations, association mining only needs to find frequent closed itemsets and their corresponding rules. An itemset*X*is a closed frequent itemset in set *S* if *X* is both closed and frequent in *S*. An important implication is that mining frequent closed itemsets has the same power as mining the complete set of frequent itemsets, but it will substantially reduce redundant rules to be generated and increase both effciency and effectiveness of mining.

Let's examine a simple example. Suppose a database contains only two transactions, "{(a1, a2, . .. , a100), (a1; a2; : : : ;a50)}", the minimum support threshold is 1 (i.e., every occurrence is frequent), and the minimum confidence threshold is 50%. The traditional association mining method willgenerate 2^{100}- 1 ≈ 10^{30}frequent itemsets, which are (a1), . . . , (a100), (a1; a2), . . . , (a99; a100), . . . ,(a1; a2; : : :; a100), and a tremendous number of association rules, whereas a frequent closed itemset mining will generate only two frequent closed itemsets: {(a1; a2; : : :; a50), (a1; a2; : : :; a100)}, and one association rule, "(a1; a2; : : :; a50) → (a51; a52; : : :; a100)", since all the others can be derived from this one easily. Pasquier et al. [5] propose an Apriori-based mining algorithm, called A-close. Zaki and Hsiao [10] propose another mining algorithm, CHARM, which improves mining efficiency by exploring an item-based data structure. According to our analysis, A-close and CHARM are still costly when mining long patterns or with low minimum support thresholds in large databases. As a continued study on frequent pattern mining without candidate generation [10], we propose an efficient method for mining closed itemsets. Three techniques are developed for this purpose: (1) the framework of a recently developed efficient frequent pattern mining method, FP-growth [10], is extended, (2) strategiesare devised to reduce the search space dramatically and identify the frequent closed itemsets quickly, and (3) a partition-based projection mechanism is established to make the mining efficient and scalable for large databases. Our performance study shows that CLOSET is efficient and scalable over large databases, and is faster than the previously proposed methods.

2. Literature Review

Mining frequent closed itemsets with projected database: An example

Let's examine how to mine frequent closed itemsetsusing the following example. Example (CLOSET) For the same transaction database TDB in Table 1 with min_sup = 2, weintroduce a divide-and-conquer method for mining frequent closed itemset. The method explores the concepts of projected database [5,6], 1. Find frequent items. Scan TDB to find the set of frequent items and derive a (global) frequentitem list, called f_list, and f_list = {c : 4, e : 4, f : 4, a : 3, d : 2}, where the items are sortedin support descending order, and the number after ":" indicates the support of the item. Foreasier understanding, the frequent items in each transaction are listed in Figure 1 according to theorderoff_list and any infrequent item, such as b, is omitted. For example, abe is listed as ea.

2. Divide search space. All the frequent closed itemsets can be divided into 5 non-overlap subsets based on the f_list: (1) the ones containing item d, (2) the ones containing item a but no d, (3) the ones containing item f but no a nor d, (4)

the ones containing e but no f, a nor d, and (5) the one containing only c. Once all subsets are found, the complete set of frequent closed itemsets is done.

3. Find subsets of frequent closed itemsets. The subsets of frequent closed itemsets can be mined by constructing corresponding conditional databases and mine each recursively.

(a) Find frequent closed itemsets containing d. Only transactions containing d are needed. The d-conditional database, denoted as $TDB|_d$, contains all the transactions having d, which is {cefa, cfa}. Notice that item d is omitted in each transaction since it appears in every transaction in the d-conditional database. The support of d is 2. Items c, f, and a appear twice respectively in $TDB|_d$. That is,every transaction containing d also containsc, f, and a. Moreover, e is infrequent since it appears only once in $TDB|_d$. Therefore, cfad : 2 is a frequent closed itemset. Since thisitemset covers every frequent item in $TDB|_d$, the mining of $TDB|_d$ finishes.

(b) Find frequent closed itemsets containing a butno d. Similarly, the a-conditional database,$TDB|_a$ = {cef, e, cf}. Item d in such transactions are omitted, since all frequent closed itemsets containing d have been found in $TDB|_d$. Since sup(a) = 3 and there is no any item appearing in every transactions in the a-conditional database, a : 3 is a frequent closed itemset. To find the remaining frequent closed itemsetscontaining a but no d, we need to further project the a conditional database. First, the set of frequent items in the a-conditionaldatabase forms a local frequent item list, f_list_a= <c : 2, e : 2, f : 2>[3]. Local infrequent item is ignored even if it is in global f_list. According to f_list_a, the frequent closed itemsetscontaining a but no d can be further partitioned into three subsets: (1) the ones containing afbutno d, (2) the ones containing aebut not d or f, and (3) the ones containing ac but no d, e or f. They can be mined by constructing conditional databases recursively. support of fa equals to that of cfad, which is a super set of faand also a frequent closed itemset already found. That means every transaction containing famust also contain cfad. Therefore, there is no frequent closed itemset containing fa but no d. Similarly, there is no frequent closed itemset containing cabut not d, e or f, since ca is a subset of cfad and sup(ca) = sup(cfad). The ea-conditional database, $TDB|_e$a = {c}, cannot generate any frequent items. Thus, ea : 2 should be a frequent closed itemset. (c) Find frequent closed itemsets containing f but no a nor d. The f-conditional database, $TDB|_f$= {ce : 3, c}, where ce : 3 indicates that cehappens three times. Since c happens in every transaction in the f-conditional database, and cf is not a subset of any frequent closed itemsetwith the same support, cf: 4 is a frequent closed itemset. Since the support of fc also equals to those of f and c, f and c always happen together, so there is no frequent closeditemsets containing c but no f. Also, that cef : 3 is not a subset of any itemset found, so it is a frequent closed itemset. (d) Find frequent closed itemsets containing e but no f, a nor d. Similarly, the e-conditional database, $TDB|_e$ = {c : 3}. But ceis not a closed itemset since it is a proper subset of cef and sup(ce) = sup(cef). However, e : 4 is a frequent closed itemsets. (e) Find frequent closed itemsets containing only c. In Step 3c, we know that there is no frequent closed itemsets containing c but no f, so there is no frequent closed itemsets containing only c. 4. In summary, the set of frequent closed itemsets found is {acdf : 2, a : 3, ae : 2, cf : 4, cef: 3, e : 4}

Table 1: The transaction database TDB

Transaction ID	Items in transaction
10	a; c; d; e; f
20	a;b;e
30	c;e;f
40	a;c;d;f
50	c;e;f

Algorithm 1 (CLOSET): Mining frequent closed itemsets by the FP-tree method.
Input: Transaction database TDB and support threshold min_sup;
Output: The complete set of frequent closed itemsets;
Method:

1) Initialization. Let FCI be the set of frequent closed itemset. Initialize FCI $\leftarrow\theta$;
2) Find frequent items. Scan transaction database TDB, compute frequent itemlistf_list;
3) Mine frequent closed itemsets recursively. Call CLOSET(θ, TDB, f_list, FCI).

3. Problem Study

3.1. Need of Frequent Itemset Mining

Studies of Frequent Itemset (or pattern set) Mining is acknowledged in the data mining field because of its broad applications in mining association rules, correlations, and graph pattern constraint based on frequent patterns, sequential patterns, and many other data mining tasks. Efficient algorithms for mining frequent itemsets are crucial for mining association rules as well as for many other data mining tasks. The major challenge found in frequent pattern mining is a large number of result patterns. As the minimum threshold becomes lower, an exponentially large number of itemsets are generated. Therefore, pruning unimportant patterns can be done effectively in mining process and that becomes one of the main topics in frequent pattern mining. Consequently, the main aim is to optimize the process of finding patterns which should be efficient, scalable and can detect the important patterns which can be used in various ways clusters and lots of additional of which the mining of association rules is one amongst the foremost widespread problems. the first motivation for looking out association rules came from the necessity to investigate thus referred to as grocery store dealing data, that is, to look at client behavior in terms of the chips (80%)" states that four out of 5 customers that bought beer conjointly bought chips. Such rules are often helpful for selections

3.2 Problem Definition

Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of items. AnitemsetX is a non-empty subset of I. For brevity, itemsetX = $\{i_{j1}, i_{j2}, \ldots, i_{jm}\}$ can also be denoted as $X = i_{j1}\, i_{j2} \ldots i_{jm}$. An itemset with m items is called an m-itemset. Duple <tid,X> is called a transaction if tid is a transaction identifier and X is an itemset. A transaction database TDB is a set of transactions.

An itemset X is contained in transaction <tid, Y>if X ⊆Y . Given a transaction database TDB, the support1 of an itemset X, denoted as sup(X), is the number of transactions in TDB which contains X. An association rule R : X → Y is an implication between two itemsets X and Y where X, Y c I and X ⋂ Y = ϕ. The support of the rule, denoted as sup(X →Y), is defned as sup(X U Y). The confidence of the rule, denoted as conf(X →Y), is defined assup(XUY)/sup(X) . As discussed by many studies, given a transaction database TDB, a minimal support thresh- old min sup, and a minimal confidence threshold min conf, the problem of association rule mining is to find the complete set of association rules in the database with support and confidence passing the thresholds, respectively. A pattern set is called a frequent pattern setif its support is no less than min sup.

4. Proposed Method

MinRPset Algorithm

Let F be the set of frequent patterns in a dataset D with respect to threshold *min_sup*, and $\hat{F}$ be the set of patternswith support no less than $min\_sup \cdot (1-\epsilon)$ in D. Obviously,$F \subseteq \hat{F}$. Given a pattern $X \in \hat{F}$, we use $C(X)$ to denote the set of frequent patterns that can be _-covered by X. We have $C(X) \subseteq F$. If X is frequent, we have $X \in C(X)$. A straightforward algorithm for finding a minimum representative pattern set works as follows. First we mine all patterns in $\hat{F}$, and then we generate $C(X)$—the set of frequent patterns that X covers—for every pattern $X \in \hat{F}$. We get$|\hat{F}|$ sets. The elements of these sets are frequent patterns in F. Let $S = \{C(X) | X \in \hat{F}\}$. Finding a minimum representative pattern set is now equivalent to finding a minimum number of sets in S that can cover all the frequent patterns in F. This is a set cover problem, and it is NP hard. We use the well-known greedy algorithm to solve the problem, which achieves an approximation ratio of $\sum_{i=1}^{k}(1/i)$,where$k$is the maximal size of the sets in S. We call this simple algorithm MinRPset. The greedy algorithm is essentially the best-possible polynomial time approximation algorithm for the set cover problem. Our experiment results have shown that it usually takes little time to finish. Generating $C(X)$s is the main bottleneck of the MinRPset algorithm when F and $\hat{F}$ are large because we need to find $C(X)$s over a large F for a large number of patterns in $\hat{F}$. We use the following techniques to improve the efficiency of MinRPset: 1) consider closed patterns only; 2) use a structure called CFP-tree to find $C(X)$s efficiently; and 3) use a light-weight compression technique to compress $C(X)$s.The number of frequent closed patterns can be orders of magnitude smaller than the total number of frequent patterns. Consider only closed patterns improves the efficiency of the MinRPset algorithm in two aspects. On one hand, it reduces the size of individual $C(X)$s since now they contain only frequent closed patterns. On the other hand, it reduces the number of patterns whose $C(X)$ needs to be generated as now we need to generate $C(X)$s for closed patterns only.

4.1 Considering Closed Patterns Only

A pattern is closed if it is more frequent than all of its supersets. If a pattern $X1$ is non-closed, then there exists another pattern $X2$ such that $X1 \subset X2$ and $supp(X2) = supp(X1)$.

Lemma 1.*Given two patterns X*1 *and X*2 *such that X*1 ⊆*X*2 *and supp(X*1*) = supp(X*2*), if X*2 *is €-covered by a pattern X,then X*1 *must be €-covered by X too.*
It implies that instead of covering all frequent patterns, we can cover frequent closed patterns only, which leads to the following lemma.

Lemma 2.*Let F be the set of frequent patterns in a dataset D with respect to a threshold min_sup. If a set of patterns R €- covers all the frequent closed patterns in F, then R €-covers*
all the frequent patterns in F.

Lemma 3.*Given two patterns X*1 *and X*2 *such that X*1 ⊆*X*2 *and supp(X*1*) = supp(X*2*), if a pattern X is €-covered by X*1*,then X must be €-covered by X*2 *too.*
This lemma also directly follows from Definition 2. It suggests that we can use closed patterns only to cover all frequent patterns. The number of frequent closed patterns can be orders of magnitude smaller than the total number of frequent patterns. Consider only closed patterns improves the efficiency of the MinRPset algorithm in two aspects. On one hand, it reduces the size of individual $C(X)$s since now they contain only frequent closed patterns. On the other hand, it reduces the number of patterns whose $C(X)$ needs to be generated as now we need to generate $C(X)$s for closed patterns only.
Algorithm MinRPset Algorithm

Description:
1: Mine patterns with support ≥ $min\_sup \cdot (1-\epsilon)$ and store them in a CFP-tree;
2: DFS_Search_CXs(*root*);
3: Remove non-closed entries from $C(X)$s;
4: Apply the greedy set cover algorithm on $C(X)$s to find representative patterns and output them;

When ϵ=0, the representative patterns are closed frequent patterns[14].

5. Result and Discussion

The proposed technique generates fewer number of closed frequent patterns than the previous algorithms..MinRPsetadditional benefits besides producing fewer representative patterns:

Users may not know what value should be used for *min_sup*at the beginning. The post-processing strategy allows users to try different *min_sup*values without mining frequent patterns multiple times. This is especially beneficial on very large datasetsAs number of dataset increases, time also increases Graph shows as the size of dataset doubles, times approximately doubles.

6. Conclusion

Mining complete set of itemsets often suffers from generating a very large number of itemsets and association

rules. Mining frequent closed itemsets provides an interesting alternative since it inherits the same analytical power as mining the whole set of frequent itemsets but generates a much smaller set of frequent itemsets and leads to less and more interesting association rules than the former. In this paper, we proposed MinRPset algorithm for locating minimum frequent pattern sets is introduced. It mines frequent patterns, and then find closed frequent patterns during a post-processing step. As a result of the utilization of the post-processing strategy, MinRP set have the extra benefits besides giving fewer representative patterns.

References

[1] J.J. Cameron, A. Cuzzocrea, F. Jiang, C.K.-S. Leung, Mining frequent itemsets from sparse data streams in limited memory environments, in: Proceedings of the WAIM 2013, Springer, pp. 51–57.
[2] E. Çem, Ö. Özkasap, ProFID: practical frequent items discovery in peer-to-peer networks, Future Gener. Comput. Syst. 29 (6) (2013) 1544–1560.
[3] C.K.-S. Leung, C.L. Carmichael, Exploring social networks: a frequent pattern visualization approach, in: Proceedings of the SocialCom 2010, IEEE Computer Society, pp. 419–424.
[4] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal.Discovering frequent closed itemsets for association rules.In Proc. 7th Int. Conf. Database Theory (ICDT'99), pages 398{416, Jerusalem, Israel, January 1999.
[5] R. Agarwal, C. Aggarwal, and V. V. V. Prasad.A tree projection algorithm for generation of frequent itemsets.In Journal of Parallel and Distributed Computing (Special Issue on High Performance Data Mining), (to appear), 2000.
[6] R. Agrawal and R. Srikant.Fast algorithms for mining association rules.In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94), pages 487{499, Santiago, Chile, September 1994.
[7] R. Agrawal and R. Srikant.Mining sequential patterns. In Proc. 1995 Int. Conf. Data Engineering (ICDE'95), pages 3{14, Taipei, Taiwan, March 1995.
[8] R. J. Bayardo. E_ciently mining long patterns from databases. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98), pages 85{93, Seattle, Washington, June 1998.
[9] J. Han, G. Dong, and Y. Yin.E_cient mining of partial periodic patterns in time series database. In Proc. 1999 Int. Conf. Data Engineering (ICDE'99), pages 106{115, Sydney, Australia, April 1999.
[10] J. Han, J. Pei, and Y. Yin.Mining frequent patterns without candidate generation. In Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00), Dallas, TX, May 2000.
[11] H. Mannila, H. Toivonen, and A. I. Verkamo.Effcient algorithms for discovering association rules.In Proc. AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94), pages 181{192, Seattle, WA, July 1994.
[12] H. Mannila, H Toivonen, and A. I. Verkamo.Discovery of frequent episodes in event sequences.Data Mining and Knowledge Discovery, 1:259{289, 1997.
[13] M. J. Zaki and C. Hsiao. Charm: An effcient algorithm for closed association rule mining. In Technical Report 99-10, Computer Science, Rensselaer Polytechnic Institute, 1999.
[14] A Flexible Approach to Finding Representative Pattern Set, Guimei Liu, Haojun Zhang, and LimsoonWongs, IEEE Transactions on knowledge and data engineering, vol. 26, no. 7, july 2014

Authors Profile

JaveriyaNaaz I. Syed has completed her Bachelor of Engineering Degree in Information Technology from S.G.B.A.U. Amravati and currenty she is pursuing her Master Degree in Engineering in Computer Science and Engineering from S.G.B.A.U. Amravati. Her area of interest is Data Mining.

Mrs. Rajeshri R. Shelke is an Asst. Professor in department of Computer Science and Engineering at Shree HVPM's COETAmravati, Maharashtra, India. Her area of interest is Data Mining.