ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 7, July 2015

A Study on Market Basket Analysis using Data-set Condensing and Intersection Pruning

Mr.Murlidher Mourya[1], Mr. J. Phani Prasad[2]

Assistant Professor, CSE, Vardhaman College of Engineering, Shamshabad, Telangana[1,2]

Abstract: Mining of frequent item sets is one of the most fundamental problems in data mining applications. A typical example is Market Basket analysis. In this method or approach it examines the buying habits of the customers by identifying the frequent items purchased by the customers in their baskets. This helps to increase in the sales of a particular product. This paper mainly focuses on the study of the existing data mining algorithm for Market Basket data. DCIP algorithm uses data-set condensing and intersection pruning to find the maximal frequent item set. The condensing process is performed by deleting items in infrequent 1-itemset and merging duplicate transactions repeatedly; the pruning process is performed by generating intersections of transactions and deleting unneeded subsets recursively. This algorithm differs from all classical maximal frequent item set discovering algorithms; experiments show that this algorithm is valid with moderate efficiency; it is also easy to code for use in KDD applications.

Keywords: Association Rule Mining, Market Basket Analysis, Mining frequent item sets, intersection pruning, and data-set condensing.

I. INTRODUCTION

Association rules can be mined and this process of mining the association rules is one of the most important and powerful aspect of data mining. One of the main criteria of ARM is to find the relationship among various items in a database. An association rule is of the form A→B where A is the antecedent and B is the Consequent. and here A and B are item sets and the underlying rule says us purchased by the customers who purchase A are likely to purchase B with a probability percentage factor as %C where C is known as confidence such a rule is as follows:

"Seventy per cent of people who purchase beer will also like to purchase diapers" This helps the shop managers to study the behavior or buying habits of the customers to increase the sales. Based on this study items that are regularly purchased by the customers are put under closed proximity. For example persons who purchase milk will also likely to purchase Bread.

The interestingness measures like support and confidence also plays a vital role in the association analysis. The support is defined as percentage of transactions that contained in the rule and is given by Support = (# of transactions involving A and B) / (total number of transactions). The other factor is confidence it is the percentage of transactions that contain B if they contain A Confidence = Probability (B if A) = P (B/A)

Confidence = (# of transactions involving A and B) / (total number of transactions that have A).

Consider the following example as

Customer	Item Purchased	Item Purchased
1	Burger	Coke
2	puff	Mineral water
3	Burger	Mineral water
4	Puff	Tea

If A is "purchased Burger "and B is "purchased mineral water" then

Support=P (A and B) =1/4

Confidence=P (B/A) =1/2

Item sets that satisfy minimum support and minimum confidence are called strong association rules.

II. RELATED WORK

PRELIMINARIES

First we provide some useful definitions

Let product set {puff, tea, soda, pizza, beer, burger, mineral water ,salad, coke, ice-cream} are represented as {I1,I2,I3,I4.I5,I6,I7,I8,I9,I10} respectively.

Boolean database

Let D = {t1 . . . tN} be a collection of Boolean tuples over the product={a1 . . . aM}, where each tuple t is a bit-vector where a 0 implies the absence of a product in the basket and a 1 implies the presence of a product in the basket.

Table 1: Database

ID	Items
01	I1, I2, I4, I5, I7
02	I1, I2, I5, I6, I7
03	I10, I3, I5, I7
04	I10, I3, I8
05	I1, I2, I3, I4, I7
06	I2, I3, I7, I8
07	I3, I6, I9
08	I1, I3, I5, I9
09	I1, I2, I6
10	I3, I4, I8, I9

III. PROPOSED TECHNIQUE: DCIP ALGORITHM

The first step of DCIP algorithm is to reduce the length of itemsets and the volume of data-set.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 7, July 2015

According to Lemma 1, any maximal frequent itemset is also a maximal frequent itemset corresponding to one transaction in D, so find all maximal frequent itemsets correspond to every transaction through intersection pruning, merge them into one set (denoted as FS hereinafter), then delete all infrequent maximal itemset in FS, and the remaining set is maximal frequent itemset. The two main processes are described as follows.

A. Condensing the Data-set

This process first sorts the data-set with descending order according to the length of its itemsets, then moves those high-dimensional transactions whose support are bigger than minimal support threshold to a frequent itemset, and deletes all subsets of those transactions to condense the data-set. These steps are as follows:

Step 1: Scan the data-set, finding all frequent 1-itemset;

Step 2: Scan the data-set, deleting all items infrequent 1-itemset from all transactions; then add up identical transactions(i.e., if transaction T1=T2, let support(T1) =support(T1) + support(T2), and delete T2 from data-sets). Sorting the data-set descendingly according to the length of itemsets to form a new data-set which we denote as C;

Step 3: Process every transaction Ti in C whose support are bigger than minimal support threshold: move T_i to FS and delete all T_j ($T_j \subset T_i$, j>i);

Step 4: Delete non-MFI from FS;

Step 5: End.

B. Intersection Pruning

Any maximal frequent itemset is also the maximal frequent itemset corresponding to a certain transaction in D; merge all maximal frequent itemset corresponding to every transaction into one set (which we denote as FS), then delete all non-frequent maximal itemsets in FS, and the remaining set is the maximal frequent itemset. These steps are as follows:

Assume we have a data-set denoted as D, and the minimal support threshold is S.

Step 1: Condense data-set D using the method described in 3.1; if |D|<S, terminate the processing for the current data-set;

Step 2: Find intersection of T_1 and T_i($1<i\leqslant n$); merge all intersections into a new data-set D1; establish the vertical data format of D; delete transaction T_i ($T_i \subset T_1$); if $|D1| \geqslant S$, then go to step 1 to perform another intersection pruning circle for D1;

Step 3: Use the vertical data format of D to find the intersection of T_j and T_i (j=2, 3, 4, ..., m<n; $j<i\leqslant n$), merge all intersections into a new data-set D1, go to step 1 to perform another intersection pruning circle for D1; when the volume of the remaining data-set is less than S, stop finding intersections of T_j and T_i, terminate the process for current data-set.

Step 4: End;

Note: Data-set condensing can be performed at the beginning of the intersection pruning process, as well as in the process of step 3.

C. Instance Analysis

The following example shows how to discover MFI using DCIP for transaction database D (Table I) with minimum support threshold as 4 (i.e., minsup=4).

TABLE I: TRANSACTION DATA-SET D

TID	Items
01	I1, I2, I4, I5, I7
02	I1, I2, I5, I6, I7
03	I10, I3, I5, I7
04	I10, I3, I8
05	I1, I2, I3, I4, I7
06	I2, I3, I7, I8
07	I3, I6, I9
08	I1, I3, I5, I9
09	I1, I2, I6
10	I3, I4, I8, I9

Step 1: Condense transaction data-set D using the method in 3.1, the result is shown in Table II;

TABLE II: RESULT OF CONDENSED D

TID	Items	Count	del
1	I1,I2,I5,I7	2	
2	I1,I2,I3,I7	1	
3	I3,I5,I7	1	
4	I2,I3,I7	1	1
5	I1,I3,I5	1	
6	I1,I2	1	1

Attribute Count is the count of corresponding transactions; attribute del indicate whether the corresponding transaction can be ignored in later processing, for example, after step 2, T6 can be ignored.

Step 2: Find intersections of T_1 and T_i (i=2, 3... 7), merge all intersections into data-set D1, as shown in Table III:

TABLE III: INTERSECTION DATA-SET FOR T1 IN TABLE II

TID	items	Count	del
1	I1,I2,I7	1(+2)	
2	I5,I7	1(+2)	
3	I2,I7	1(+2)	1
4	I1,I5	1(+2)	
5	I1,I2	1(+2)	1

Establish vertical data format for D; because in Table II, $T_6 \subset T_1$, T_6.del=1 (see Table II) . The (+2) for attribute Count in table III is the count of T_1 in Table II.

Step 3: Condense the data-sets in Table III; as this example, the result remains no change.

Step 4: Find intersections of T_1 and T_i (i=2, 3, 4, 5) in Table III respectively, merge them into a new data-set D1, as shown in Table IV.

TABLE IV: INTERSECTION DATA-SET FOR T1 IN TABLE III

TID	items	Count	del
1	I2,I7	1(+2+1)	
2	I1,I2	1(+2+1)	

Because T_3 and T_5 are subset of T_1 in Table III, delete T_3 and T_5;

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 7, July 2015

Step 5: Condense the data-set in Table IV, produce frequent itemset {{I2, I7}:4, {I1, I2}:4}; Table IV is now empty after condensing;

Step 6: Back to Table III, T_3 and T_5 has been deleted, we only need to find the intersection of T_2 and T_4; but the length of T_2 and T_4 are both 2, no need to find intersection of them.

Step 7: Back to Table II, because T_6 has been deleted, we only need to find the intersections of T_2 and T_i (i=3, 4, 5); merge all intersections into a new data-set D1, as shown in Table V.

TABLE V: INTERSECTION DATA-SET FOR T2 IN TABLE 2

TID	items	Count	del
1	I3,I7	1(+1)	
2	I2,I3,I7	1(+1)	
3	I1,I3	1(+1)	

Because $T_4 \subset T_2$ in Table II, it should be deleted.

Step 8: Condense the data-set in Table V; after Condensing the result is empty;

Step 9: The original data- set D has 10 transactions; Table II shows that 30% (3 transactions) of them has been processed; condense again the remaining data-set in Table II, and the result is empty. The process ends.

Step 10: Merge all resulting frequent item sets, and delete all non-frequent maximal item sets, the final result of MFI is {{I2, I7}: 4, {I1, I2}: 4}.

The steps above use 14 times of intersection calculations for MFI; compared with other Apriori-like algorithms, its simplicity and efficiency is explicit.

Note: Because the volume of the example data-set D is small (only 10), the above process does not include the utilizing of vertical data format; the reason of introducing vertical data format is to reduce the number of times of finding the intersections.

IV. PERFORMANCE STUDY

If the length of the longest transaction item set is L, the depth of recursive calling of the algorithm itself is less than L-2. The number of calculations for intersections is negatively correlated with support, number of deleted transactions, and number of duplicated transactions. This algorithm can also be implemented parallels for each transaction's maximal frequent itemset to get more efficiency. It is valid for both long and short frequent pattern mining applications; for vast volume of data-set, its usability retains because of the time & space cost increases not very drastically.

Another advantage of DCIP algorithm is its easy implementation. It is coded and tested using PowerBuilder script language on a microcomputer with Pentium IV/1.80GHz CPU, 512M memory running Windows XP operation system. Testing dataset is extracted from a supermarket's sales record.3000, 5000, 10000 and 20000 transactions are tested respectively with each record having 2-10 categories of commodity (the average number of categories is 6). Figure 1 shows the running time for different volume of datasets with minimum support threshold of 5%, 20% and 50%respectively. The bigger minimum support threshold, the lesser time needed for MFI.

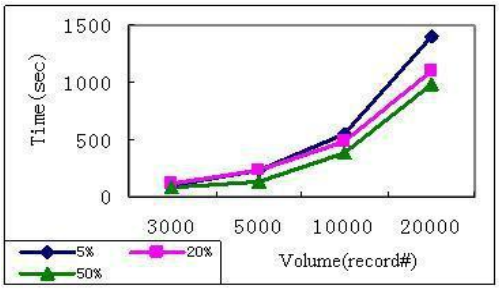

Figure 1:Performance test for multiple data-set &supports

V. CONCLUSION

DCIP provides a new and efficient algorithm for discovering MFI; it condenses data-set by deleting items in infrequent 1-itemsets and merging duplicate transactions repeatedly, and utilizes the intersections of (1-s)*|D|+1 transactions with other transaction item sets to perform pruning; along with the discovering process, with the increasing of the number of deleted transactions, the number of times needed for calculating intersections will decrease rapidly. It's time & space cost increases not drastically when data-set volume increases, so its usability retains for MFI applications for high volume data-sets.

The DCIP algorithm can be further optimized in various aspects, such as keep a record of all resulting intersections to avoid duplicated generation of identical intersections to further improve the efficiency of this algorithm. While the problems considered in this paper are novel and important to the area of ad hoc data exploration and retrieval, we observe that our specific problem definition does have limitations. After all, a query log is only an approximate surrogate of real user preferences, and moreover, in some applications neither the database, nor the query log may be available for analysis; thus, we have to make assumptions about the nature of the competition as well as about the user preferences.

REFERENCES

[1]*D. Burdick, M. Calimlim, and J. Gehrke* (2001)"MAFIA: A Maximal Frequent Item Set Algorithm for Transactional Databases," Proc. Int'l Conf. Data Eng. (ICDE), 2001.

[2]*M.R. Garey and D.S. Johnson* (1979), "Computers and Intractability:A Guide to the Theory of NP-Completeness".

[3]W Yanthy, T. Sekiya, K. Yamaguchi , "Mining Interesting Rules by association and Classification Algorithms", FCST 09.

[4]Chiu, K.S.Y., Luk, R.W.P, Chan, K.C.C., and Chung, K.F.L, "Market-basket Analysis with Principal Component Analysis:An Exploration", IEEE International Conference on Systems, Man and Cybernetics, Vol.3, 2002.

[5]Cunningham , S.J. and Frank, E., "Market Basket Analysis of Library Circulation Data", International Conference on Neural Information Processing, Vol.2. 1999.

DOI 10.17148/IJARCCE.2015.4760