Mining Maximal Frequent Patterns With Similarity Matrices of Data Records

Hua Yuan[1] **Junjie Wu**[2]

[1]Dept. of Management Science and E-commerce,
University of Electronic Science and Technology of China, Chengdu 610054, China
[2]School of Economics and Management,
Beihang University, Beijing 100083, China

Abstract

In this paper, we proposed a similarity matrix based method to mining maximal frequent patterns from large database. The study is very different from the previous Apriori-liked method. Especially, the method can be performed directly on the original data in database without various format transformation. The analyzing and experimental results show that the method is useful for frequent pattern mining tasks with large data set.

Keywords: Data mining; Maximal frequent pattern; Similarity matrix;

1. Introduction

In frequent pattern mining, most of the proposed mining algorithms are a variant of Apriori, which employs a bottom-up, breadth-first search to get all the itemsets. In many applications (especially in dense data) with long frequent patterns enumerating all possible subsets is computationally unfeasible[1]. On the other hand, some applications do not necessary to find out all the itemsets, for example, gene expression data used in bioinformatics[2], program trace data used in software engineering research[3]. Since the maximal frequent itemsets contain all the frequent ones, thus, there has been recent interest in mining maximal frequent patterns[4, 1].

DepthProject[4] finds long itemsets using a depth-first search of a lexicographic tree of itemsets. MaxMiner[5] employs a breadth-first traversal of the search space; it reduces database scanning by employing a lookahead pruning strategy. Mafia[6] uses a few pruning strategies to remove non-maximal sets, and uses vertical bit-vector data format, and compression and projection of bitmaps to improve performance. GenMax[1] is a backtrack search based algorithm, and uses a number of optimizations to prune the search space and is good at returning the exact set of maximal frequent itemsets. From the perspective of counting the number of solutions, Yang[7] studied the complexity-theoretic aspects of maximal frequent itemset mining. In [8], Hasan and Zaki proposed a new approach to mining frequent pattern representatives based on a uniform sampling of the output space.

Since the frequent patterns mining is to find the patterns that appear

frequently in a sample data set, so the appearances of records that contain the same pattern would look similar to each other. In this paper, we proposed a similarity matrix based method to mining maximal frequent patterns from large database.

The remaining of the paper is organized as follows. Section 2 describes the preliminaries; Section 3 presents the algorithm of maximal frequent pattern mining with similarity matrix; Some experimental results are presented in Section 4; Finally, Section 5 concludes this paper.

2. Preliminaries

In this section, we introduce the important concepts and notations that will be used throughout the paper.

2.1. Database and itemset

Given a data set S, we will use the notation, $|S|$, to denote the cardinality of S, i.e., the number of elements in S.

The *database* (data set) in this paper is identified with D and comprises a set of records (transactions). The total number of records is $|D|$. Each record t in database D has a unique identifier, *id*, and contains a set of items. Note that even if two database records (transaction) contain the same set of items they are still different from each other, since each record has its own unique *id*. Let t_{id} denote the *id*-th record, then for any $1 \leq i, j \leq |D|$ and $i \neq j$, records $t_i \neq t_j$.

A set of items is often called an *itemset*. Let $I = \{\mathrm{i}_1, \mathrm{i}_2, ..., \mathrm{i}_n\}$ be an itemset with n items, then the notation, $I \subseteq t$, to denote that I is a subset of the set of items that t contains. We will use the notation, $D(I)$, to represent the set of records (transactions) of D that are a superset of I, i.e.,

$$D(I) = \{t | I \subseteq t, t \in D\}. \quad (1)$$

2.2. Maximal frequent itemset

The support of an itemset I, denoted $\sigma(I)$, is the number of transactions in which that itemset occurs as a subset. An itemset I is frequent if its support is more than or equal to some user specified threshold value, minimum support (*minsupp*), i.e., if $\sigma(I) \geq minsup$. A frequent itemset is called *maximal* if it is not a subset of any other frequent itemset[1].

2.3. Similarity matrix

To measure the similarity of two records, a record from D is needed to choose as the *base record* for the similarity comparison, then we introduce a *similarity vector* to keep the comparison results. The *similarity vector* in this paper is a vector in which each entry has value either 0 or 1 and is defined as follows.

Definition 1. *(Similarity Vector)*
Assume database D has $|D|$ records, $t_1, t_2, ..., t_{|D|}$, and maximum n fields. By setting t_i as the the base record, the similarity between $t_i = \{i_{i1}, i_{i2}, ..., i_{ir}\}$, $r \leq n$, and $t_j = \{i_{j1}, i_{j2}, ..., i_{js}\}$, $s \leq n$, is denoted by a vector, $v(t_i, t_j)$, which is defined as follows.

$$v(t_i, t_j)[p] = \begin{cases} 1, & \text{if } i_{ip} = i_{jq}, \\ 0, & \text{Otherwise.} \end{cases}$$

where $p = 1, ..., r, q \in [1, s]$.

Definition 2. *(Item Projection)*
Given an itemset $\boldsymbol{a}$ with r items and a vector $\boldsymbol{b}$ with r elements of 1 or 0, the item projection operation is defined as

$$Proj_{\boldsymbol{b}}\boldsymbol{a} = \boldsymbol{a} \cdot \boldsymbol{b} = \{a_j \cdot b_j\}, j = 1...r, \quad (2)$$

where

$$a_j \cdot b_j = \begin{cases} a_j, & \text{if } b_j = 1; \\ \phi, & \text{if } b_j = 0. \end{cases}$$

The (0-1) vector $\boldsymbol{b}$ in definition 2 is the *implication pattern* of itemset $\boldsymbol{a}$. That is, the different formation of vector $\boldsymbol{b}$ can be used to represent the various patterns in $\boldsymbol{a}$.

Theorem 1. *Two important properties:*

(1) The result of item projection operation is a subset of $\boldsymbol{a}$*, i.e.,* $\boldsymbol{a} \cdot \boldsymbol{b} \subseteq \boldsymbol{a}$.

(2) Choosing record t_i *as the base record of similarity comparison, the common pattern of* t_i *and* t_j *can be calculated as*

$$\boldsymbol{x}_{ij} = t_i \cdot v(t_i, t_j). \qquad (3)$$

then, $\boldsymbol{x}_{ij} = \boldsymbol{x}_{ji}$.

Consider the example data set[1] in following Table 1.

Table 1: Data set.

TID	Items
1	ACTW
2	CDW
3	ACTW
4	ACDW
5	ACDTW
6	CDT

Table 2: SM_1.

t_1	$[A\ C\ T\ W]$
$v(1,1)$	[1 1 1 1]
$v(1,2)$	[0 1 0 1]
$v(1,3)$	[1 1 1 1]
$v(1,4)$	[1 1 0 1]
$v(1,5)$	[1 1 1 1]
$v(1,6)$	[0 1 1 0]

According to the definition of similarity vector, we obtain that $v(1,2) =$ [0 1 0 1] and $v(1,3) =$ [1 1 1 1]. Note that the symmetry property of v, i.e. $v(t_i, t_j) = v(t_j, t_i)$, is not always holds in nature. Fox example, in Table 1, $v(2,1) =$ [1 0 1] $\neq v(1,2)$, but $v(3,1) = v(1,3)$. Moreover, the pattern supported by transaction 1 and 2 is $\boldsymbol{x}_{12} = t_1 \cdot v(1,2) = \{ACTW\} \cdot$ [0 1 0 1] $= \{CW\}$. Also, it can be verified that $\boldsymbol{x}_{21} = t_2 \cdot v(2,1) = \{CW\}$.

Once the *base record* is selected, a one-to-one correspondence between vectors and pair of data records can be established.All the similarity vectors of base record t_i constitute a *Similarity Matrix* of t_i, SM_i. That is

$$v(i,j)[k] = SM_i[j,k], \quad k = 1, ..., r. \qquad (4)$$

The generation of similarity matrix can be describe as algorithm 1.

Algorithm 1 Similarity matrix generation algorithm

```
1: Input: t_i = {i_i1, i_i2, ..., i_ir} ∈ D;
2: Output: Matrix SM_i[|D|, r];

3: SM_i = 0;
4: for l = 1 to |D| do
5:     for m = 1 to s do
6:         for k = 1 to r do
7:             if i_jm = i_ik then
8:                 SM_i[l, k] = 1;
9:                 break;
10:            end if
11:        end for
12:    end for
13: end for
14: return SM_i;
```

Setting record 1 in Table 1 as the base record, we can obtain the similarity matrices of SM_1 as shown in Table 2.

2.4. Support and Maximality

Frequent itemset mining is mainly concerned with those itemsets that are

present in a database frequently and the number of appearances of an itemset in a database is commonly referred to as its *support* which can be formalize as follows.

Definition 3. *(Support)*
Let I be an itemset and D a database (data set). The support of I in D is the number of transactions of D in which I appears as a subset, i.e.,

$$supp(I) = |D(I)|. \quad (5)$$

Especially, the *support* of the rth item of record t_i is:

$$supp(\{\mathrm{i}_{ir}\}) = \sum_{l=1}^{|D|} SM_i[l, r]. \quad (6)$$

Definition 4. *(Maximality)*
For each record $t_j, j \in [1, |D|]$, we define its maximum similarity (maximality) with record t_i as $maxi$.

$$maxi(t_i, t_j) = \sum_{k=1}^{r} SM_i[j, k]. \quad (7)$$

Choosing record 1 in Table 1 as the base record, the calculation results of *support* and *maximality* are shown in Table 3, where $\mathrm{i}_{11} = A$, $\mathrm{i}_{12} = C$, $\mathrm{i}_{13} = T$, $\mathrm{i}_{14} = W$. It is easy to calculate the *support* of each item in record 1 with Table 3. For example, $supp(\mathrm{i}_{11}) = 4$ means the *support* of item "A" is 4.

Obviously, the *Maximal Patterns* may lies in the patterns with bigger *maximality* and *supp* reveals their potential *frequency*.

3. Mining Maximal Frequent Patterns

In the following section, we will propose a new method along with a demonstration example to mine the *Maximal Frequent Patterns* basing on the properties of frequent pattern and similarity matrices.

Table 3: An example for *supp* and *maxi*

t_1	$[\ A\ C\ T\ W]$	$maxi(1, j)$
$v(1,1)$	[1 1 1 1]	4
$v(1,2)$	[0 1 0 1]	2
$v(1,3)$	[1 1 1 1]	4
$v(1,4)$	[1 1 0 1]	3
$v(1,5)$	[1 1 1 1]	4
$v(1,6)$	[0 1 1 0]	2
$supp(\mathrm{i}_{1r})$	4 6 4 5	

3.1. Properties about frequent pattern support

Here we will introduce some properties about frequent pattern.

Lemma 1. *Let D be a database of data records, X and Y two itemsets. If $X \subseteq Y$, then $D(X) \supseteq D(Y)$ and $|D(X)| \geq |D(Y)|$[7].*

Lemma 1 describes the common sense of the relations between the frequent itemsets and its subsets.

Theorem 2. *Given an itemset $I = \{i_1, ..., i_r\}, (1 \leq r \leq n)$ in D, the following conclusion holds:*

$$\min_{j \in [1,r]} \{supp(\{i_j\})\} \geq |D(I)|. \quad (8)$$

Proof. For any item $\mathrm{i}_j, j \in [1, r]$ from I, since that $\mathrm{i}_j \in I$, then $\{\mathrm{i}_j\} \subseteq I$. Further, with definition 3 and lemma 1 we can obtain finally the expression $supp(\{\mathrm{i}_j\}) = |D(\{\mathrm{i}_j\})| \geq |D(I)|$.
So, $\min_{j \in [1,r]} \{supp(\{\mathrm{i}_j\})\} \geq |D(I)|$ holds true. □

To speed up the calculation processes, theorem 2 enable us to discard the irrelevant items and the items with very small *support* such that $support < min\_supp$.

3.2. Methodology

There is mainly three steps for discovering the *Maximal Frequent Patterns*: Similarity matrices calculating; Extracting *Maximal Frequent Patterns* and candidate patterns fusion.

First, we need to calculate the similarity matrix of the selected base record $t_i, i \in [1, |D|]$. (See algorithm 1.)

The next is to extract *Maximal Frequent Patterns* from similarity matrices with a user specified support threshold, *min_supp*.

Since we know that the vector $v(i, j)$ in SM_i, also can be referred as $SM_i[j,]$, will be scored bigger *maximality* when t_j has high similarity with t_i, then the *Maximal Frequent Pattern* discovering process can be began with the vector in SM_i which has a maximum *maximality*. The precess is as follows.

- Find the vector $v(i, j)$ with maximum *maximality*, $j \neq i$;
- Delete the irrelevant columns from SM_i according to the zero elements of $v(i, j)$;
- Delete the columns that satisfy $support < min\_supp$ from SM_i according to the non-zero elements of $v(i, j)$;
- Delete the rows that has zero elements from $SM(i)$.

The remainder of the items in t_i is a candidate of *Maximal Frequent Pattern*. The process is shown in algorithm 2 as a whole.

Here is an example about algorithm 2 with transaction data in Table 1. It is easy to calculate that $Supp = [\ 4\ 6\ 4\ 5\ ]$ and $Maxi^T = [\ 4\ 2\ 4\ 3\ 4\ 2\ ]$. So

$$\max_{j=1,...,|D|, j\neq 1} \{Maxi[j]\} = Maxi[3] = 4,$$

and $V\_temp = SM_1[3,] = [\ 1\ 1\ 1\ 1\ ]$.

Algorithm 2 Maximal frequent pattern discovering algorithm

Input: $SM_i, i = 1...|D|$; *min_supp*;
Output: Maximal frequent pattern set MFP;

$MFP = \phi$;
$Maxi[j] = 0, j = 1, ..., |D|$;
$Supp[j] = 0, j = 1, ..., r_i$;
$V\_temp[j] = 0, j = 1, ..., r_i$;
for $i = 1$ to $|D|$ **do**
  while $Supp[] \geq min\_supp$ **do**
    $\max_{j=1,...,|D|, j\neq i}\{Maxi[j]\} = Maxi[m]$;
    $V\_temp[] = SM_i[m,]$;
    for $j = 1$ to r_i **do**
      if $Supp[j] < min\_supp$ **then**
        $V\_temp[j] = 0$;
      end if
    end for
    Compute $\boldsymbol{x}$=$t_i \cdot V\_temp$;
    Compute $supp(\boldsymbol{x})$;
    $MFP = MFP \cup \{\boldsymbol{x}\}$;
    for $j = 1$ to r_i **do**
      if item $i_{ij} \in \boldsymbol{x}$ **then**
        $Supp[j] = Supp[j] - supp(\boldsymbol{x})$;
      end if
    end for
  end while
end for
return MFP;

Assume that the user specified threshold is $min\_supp = 5$, apparently, $Supp[1]$ and $Supp[3]$ are less than the threshold, then V_temp is modified as $V\_temp = [\ 0\ 1\ 0\ 1\ ]$ and the candidate of MFP is $\boldsymbol{x}$= $t_1 \cdot V\_temp = \{CW\}$.

At the next step, the process of discovering candidates for *Maximal Frequent Pattern* on $SM_2, ..., SM_6$ an be handled in the same way.

3.3. Support of Maximal frequent pattern

Since the *Maximal Frequent Pattern* discovering process is begin with the similar vector $v(i,j)$ in SM_i which has the maximum *maximality*, we can change the element value of $v(i,j)$ as follows.

$$v_m[k] = \begin{cases} 0, & \text{if } SM_i[j,k] = 0, \text{ or} \\ & supp(\{i_{ik}\}) < min\_supp \\ 1, & \text{otherwise.} \end{cases} \quad (9)$$

With the conclusion of theorem 2, the itemset in t_i with the pattern formation that implicated in v_m is a candidate of *Maximal Frequent Pattern*. Let $\boldsymbol{x}$ denote the implicated frequent pattern, then

$$\boldsymbol{x} = t_i \cdot v_m,$$

and

$$D(\boldsymbol{x}) = \{v | v \in Proj_{v_m} SM_i \text{ and } v[k] \neq 0\}, \quad (10)$$

where $k = 1, ..., |\boldsymbol{x}|$.

With matrix SM_1 in Table 3 and $min\_supp = 5$, we can obtain that $v_m = [\ 0\ 1\ 0\ 1\ ]$ according to formulation (9). The results of $Proj_{[\ 0\ 1\ 0\ 1\ ]} SM_1$ can be seen in Table 4.

Table 4: Finding support transactions

$\boldsymbol{x}$	[C W]
$v(1,1)$	[1 1]
$v(1,2)$	[1 1]
$v(1,3)$	[1 1]
$v(1,4)$	[1 1]
$v(1,5)$	[1 1]
~~$v(1,6)$~~	~~[1 0]~~

By deleting the rows that has zero elements (sum of the row elements are less than $|\boldsymbol{x}| = |\{CW\}| = 2$) from Table 4, we obtain $supp(\{CW\}) = 5$, then $Supp = [\ 4\ 6\ 4\ 5\ ]$ is modified as $Supp = [\ 4\ 1\ 4\ 0\ ]$, all the elements in $Supp[]$ are less than min_supp, so the discovering process stopped.

4. Experiments and Discussion

4.1. Experiment setup

Benchmark Datasets: The remaining datasets used in our evaluation were taken from the UCI Machine Learning Database Repository (http://archive.ics.uci.edu/ml/). We favored those with categorically-valued attributes, relatively wide record length, a substantial number of records, and they are cited as benchmark to evaluate algorithm performance in many previous excellent work. We chose the datasets of chess (which are compiled from game state information) in the following experiments and Table 5 lists the width and height of the datasets.

Table 5: Database Characteristics

Dataset	Records	#of Attributes
chess	3, 196	36

4.2. Main results

The proposed method in this paper is a greedy algorithm in mature, the performance of efficiency is relative poor. With theorem 1 and taking *minimum support* into consideration, it is rational to infer that the last *minsupp* records can be ignored in mining process.

Figure 1 shows the performance of the time consumption of the method,

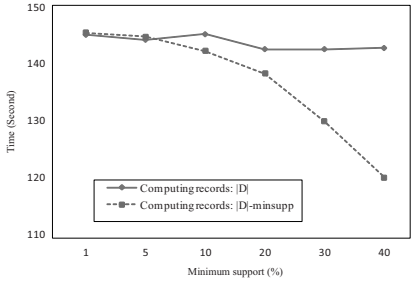

Fig. 1: Time consumption of the algorithm.

from which, we know that the computing process with all the datasets has little sensitive with the threshold of *minimum support* (the execution time become little faster with bigger hreshold), but the performance of improved algorithm was dramatically affect by the threshold.

The initial algorithm is greedy in finding frequent patterns to keep the completeness of the found results, and the total execution time is most relevant to the magnitude of the dataset, i.e. the value of $|D|$. So, in comparison with the experiments results in [1], we can find that this is why our method performs stably in finding frequent patterns with the increasing *minimum support*.

5. Conclusion

In frequent pattern mining, the records containing same pattern would look similar in appearance to each other. Along this line of consideration, we proposed a similarity matrix based method to mining maximal frequent patterns from large database. The study is very different from the previous Apriori-liked method:

- First, select a record from dataset as the base record and generate a similarity matrix with other records, the common patterns lies in the base record will appears repeatedly in the matrix.
- With the generated matrix, calculating 1) the support of each item contained in the base record, and 2) the maximality of each pattern in the base record.
- Finally, generate maximal frequent patterns with the found patterns and *minimum support* of each item.

Taking the symmetrical property of similarity (Theorem 1) into consideration, we improve the efficiency of the algorithm primitively by lessening the scanning of datasets.

The results of experiments on both exemplary data and real datasets show that the method can be use to explore maximal frequent patterns. The main deficiency of the method is the high time consumption,the future work is for enhancing the performance of the algorithm.

Acknowledgment

The research was partially supported by the National Natural Science Foundation of China (NSFC) (Nos. 70901002, 70890080), the MOE Project of Key Research Institute of Humanities and Social Sciences at Universities (No. 07JJD630005), and the Ph.D. Programs Foundation of Ministry of Education of China (No. 20091102120014).

References

[1] Karam Gouda and Mohammed J. Zaki. Efficiently mining maximal frequent itemsets. In *Proceedings of ICDM 2001*, pages 163–170, 2001.

[2] Ronnie Alves, Domingo S. Rodriguez-Baena, and Jesus S. Aguilar-Ruiz. Gene association analysis: a survey of frequent pattern mining from gene expression data. *Briefings in Bioinformatics*, 11(2):210–224, 2010.

[3] Christopher LaRosa, Li Xiong, and Ken Mandelberg. Frequent pattern mining for kernel trace data. In *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC)*, pages 880–885, 2008.

[4] Ramesh C. Agarwal, Charu C. Aggarwal, and V. V. V. Prasad. Depth first generation of long patterns. In *Proceedings of the ACM SIGKDD Conference*, pages 108–118, 2000.

[5] Roberto J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, 1998.

[6] Doug Burdick, Manuel Calimlim, and Johannes Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proceedings of the 17th ICDE*.

[7] Guizhen Yang. Computational aspects of mining maximal frequent patterns. *Theoretical Computer Science*, Volume 362, Issues 1-3:63–85, 2006.

[8] Mohammad Al Hasan and Mohammed Zaki. Musk: Uniform sampling of k maximal patterns. In *SIAM Data Mining*, pages 650–661, 2009.