An Updated & More Efficient Algorithm for Mining Sequential Patterns

[1]Krishnkant Solanky, [2]Abhishek Raghuvanshi

[1]Research Scholar, [2]HOD of CE/IT Department
Mahakal Institute of technology

***ABSTRACT*: Tremendous amount of data being collected is increasing speedily by computerized applications around the world. Hidden in the vast data, the valuable information is attracting researchers of multiple disciplines to study effective approaches to derive useful knowledge from within. Among various data mining objectives, the mining of frequent patterns has been the focus of knowledge discovery in databases. This thesis aims to investigate efficient algorithm for mining including association rules and sequential patterns. Mining sequential patterns with time constraints, such as time gaps and sliding time-window, may reinforce the accuracy of mining results. However, the capabilities to mine the time-constrained patterns were previously available only within Apriori framework. Recent studies indicate that pattern-growth methodology could speed up sequence mining.**

1 Introduction

Recent developments in computing [1] and automation technologies have resulted in computerizing business and scientific applications in diverse areas. Turing the huge amounts of accumulated data into knowledge is attracting researchers in various domains including databases, machine learning, statistics, and so on. From the perspectives of database researchers, the emphasis is on discovering useful patterns hidden within the large data sets. Hence, a central issue for knowledge discovery in databases, also the focus of this thesis, is to develop efficient and scalable mining algorithms as integrated tools for database management systems.

Data mining, which is also referred to as *knowledge discovery in databases*, has been recognized as the process of extracting non-trivial, implicit, previously unknown, and potentially useful information from data in databases . The database used in the mining process generally contains large amounts of data collected by computerized applications. For example, bar-code readers in retail stores, digital sensors in scientific experiments, and other automation tools in engineering often generate tremendous data into databases in a very fast speed. Not to mention the natively computing-centric environments like Web access logs in Internet applications. These databases thus serve as rich and reliable sources for knowledge generation and verification. Meanwhile, the large databases present challenges for effective approaches for knowledge discovery.

The discovered knowledge [2] can be used in many ways in corresponding applications. For example, identifying the frequently appeared sets of items in a retail database can be used to improve the decision making of merchandise placement or sales promotion. Discovering patterns of customer browsing and purchasing (from either customer records or Web traversals) may assist the modeling of user behaviors for customer retention or personalized services. Given the desired databases, whether relational, transactional, spatial, temporal, or multimedia ones, we may obtain useful information after the knowledge discovery process if appropriate mining techniques are used
A typical process of knowledge discovery in databases is illustrated in Fig. 1-1.

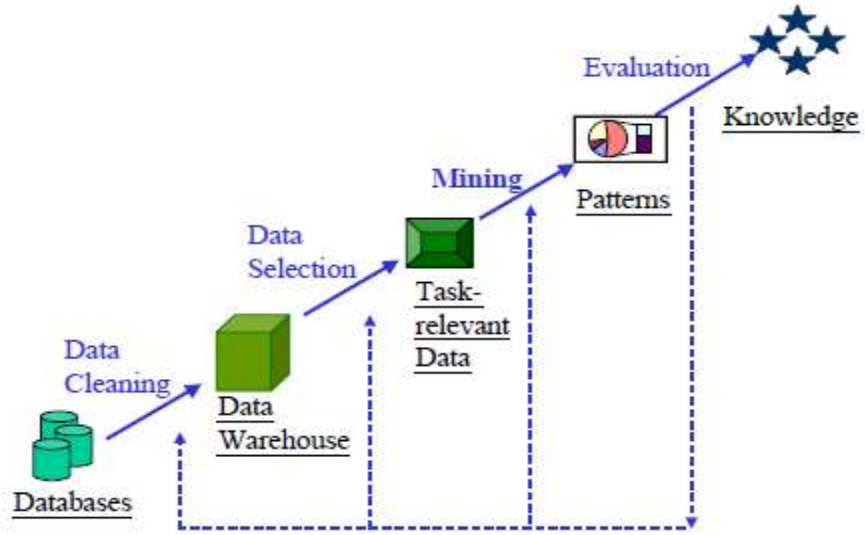

Fig. 1-1. The process of knowledge discovery in databases

Having the databases, relevant prior knowledge, and the goals of the application domain, the target data set is created by *selecting* the data required. The *data cleaning* in Fig. 1-1 may removes those 'dirty' data, e.g. data with incomplete fields, missing or wrong values, in the preprocessing stage. The 'clean' data is then reduced and/or transformed so that the data is represented by the useful features and actionable dimensions. To find the patterns of interest, the users perform the required *mining* functions, which include summarization/generalization of data characteristics, classification/clustering of data for future prediction, association finding for datacorrelation, trend and evolution analysis, etc. The discovered patterns are *evaluated* and presented as knowledge. The process may iterate and contain certain loops between any two steps.

Of all the mining functions in the knowledge discovering process, frequent pattern mining is to find out the frequently occurred patterns. The measure of frequent patterns is a user-specified threshold that indicates the minimum occurring frequency of the pattern. We may categorize recent studies in frequent pattern mining into the discovery of association rules and the discovery of sequential patterns. Association discovery finds closely correlated sets so that the presence of some elements in a frequent set will imply the presence of the remaining elements (in the same set). Sequential pattern discovery finds temporal associations so that not only closely
correlated sets but also their relationships in time are uncovered.

Considering a sequential pattern having [3,4] three items, the constitution of the pattern could be a list of: (1) three elements where each element is an item (2) two elements where the first element has one item and the second has two items (3) two elements where the first element has two items and the second has one item (4) one element that has three distinct items. Given the same number of possible items in the itemset database and the sequence database, the potential sequential patterns having three items greatly outnumber the potential itemsets having three items. The total number of candidates, which contains more than patterns having three items, increases exponentially as the number of possible item increases. Searching in the larger and more complex sequence database with the enormous number of candidates demands highly efficient mining algorithms.
Common sequence mining considers no constraints for the time-gaps between adjacent elements of a pattern, thereby introducing some uninteresting patterns at times. For example, without specifying the maximum time gap (between adjacent elements), one may discover an example pattern such as "many customers bought *LCD-projector* after purchasing *Laser-pointer*." Nevertheless, the pattern could be insignificant if the time interval between the two elements is too long such as over years. Typical time constraints include minimum gap, maximum gap, and sliding time-window [5]. In this thesis, we will look into the time-constraint problem and propose an approach that integrates these constraints for the discovery of sequential rules with time constraints.

Sequence Database Each **sequence** is an time-ordered list of itemsets. An **itemset** is an unordered set of items (symbols), considered to occur simultaneously.

ID	Sequences
seq1	$\{a, b\},\{c\},\{f\},\{g\},\{e\}$
seq2	$\{a, d\},\{c\},\{b\},\{a, b, e, f\}$
seq3	$\{a\},\{b\},\{f\},\{e\}$
seq4	$\{b\},\{f, g\}$

Sequential Pattern Mining is probably the most popular set of techniques for discovering temporal patterns in sequence databases. SPM finds subsequences that are common to more than *minsup* sequences. SPM is limited for making **predictions.** For example, consider the pattern {x},{y}. It is possible that *y* appears frequently after an *x* but that there are also many cases where *x* is not followed by *y*. For **prediction**, we need a measurement of the confidence that if *x* occurs, *y* will occur afterward .

Proposed Algorithm:
The steps are as follows

STEP 1: START
STEP 2: INPUTS ARE:
- **SEQUENTIAL DATA SET D&**
- **MINIMUM SUPPORT THRESHOLD.**

STEP 3: FIRST THE PROPOSED ALGORITHM SCANS THE SEQUENTIAL DATA BASE D AND CALULATES THE SUPPORT OF EACH SINGLE SIZE ITEM FROM D.
STEP 4: NOW ELIMINATE ALL THE INFREQUNT ITEMS FOUND IN STEP 3 FROM D SO THAT D WILL BE CONVERTED IN TO A COMPRESSED SEQUENTIAL DATA BASE.

STEP 5: ALGORITHM IS CALLED RECURSIVELY TO GENERATE BIGGER SEQUENTIAL PATTERNS BY USING THE UNION OR EXPANSION OF LOWER SIZE ITEMS.

Example:

Consider the following sequential data set with the minimum support 3

Table 1 : Sequential Data Base

Sequence ID	Sequences
S1	<(1) (1,2,3) (4) (7,8) (3)>
S2	<(3) (5,8,9) (1,2) (2,3)>
S3	<(5) (1,2) (3,5,6) (1,2) (6)>

The data set is scanned to find the sequential frequent patterns of the size 1:

1- Suppot 3
2- Support 3
3- Support3

The items 4,5,6,7,8 have support less than the 3.

In pruning step, all the infrequent items are eliminated from the original data set. Because it is clear that they will not appear in any frequent sequential pattern.

Bu doing this, the original data set is converted in to the transformed & much compressed data set. It is as follows:

Table 2

Sequence ID	Sequences
S1	<(1) (1,2,3) (3)>
S2	<(3) (1,2) (2,3)>
S3	<(1,2) (3) (1,2)>

Now expand the size 1 items to get the patterns of larger size:

For example 1 is expanded in to (1,2) & (1,3). Then these two are checked for the minimum support in table2. We see that the support is 3. So this two are also frequent sequential patterns.

The algorithm is continued until there are items to be expanded. At the end, we get following sequential patterns

(1), (2), (3), (1,2), (1,3), (2,3), (1,2,3)

Conclusion:

In this paper we presented an algorithm to quickly find all frequent sequences in a list of transactions. The algorithm utilizes a vertical bitmap representation to store each sequence. Our algorithm is fast in comparison to recusive suffix prefix algorithm

References:

[1] Agrawal, R., Imielminski, T., and Swami, A. Mining Association Rules Between Sets of Items in Large Databases, In *Proc. SIGMOD Conference*, (Washington D.C., USA, May 26-28, 1993) 207-216.

[2] Mannila, H., Toivonen and H., Verkano, A.I. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1, 1 (1997), 259-289

[3] Das., G., Lin, K.-I., Mannila, H., Renganathan, G., and Smyth, P. Rule Discovery from Time Series. In *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining* (New York, USA, August 27-31, 1998), 16-22.

[4] Harms, S. K., Deogun, J. and Tadesse, T. 2002. Discovering Sequential Association Rules with Constraints and Time Lags in Multiple Sequences. In *Proc. 13th Int. Symp. on Methodologies for Intelligent Systems* (Lyon, France, June 27-29, 2002), pp. .373-376.

[5] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," *Proceedings* of the *5th International Conference on Extending Database Technology*, Avignon, France, pp. 3-17, 1996. (An extended version is the IBM Research Report RJ 9994)

[6] J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal and M.-C. Hsu, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-projected Pattern Growth," *Proceedings of 2001 International Conference on Data Engineering*, pp. 215-224, 2001.