An Efficient Technique for Mining Sequential Patterns from Standard Data Set

Amit Sariya[1] Kshitij Pathak[2]
[2]Assistant Professor
[1, 2] MIT, Ujjain

Abstract--- 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. Current algorithms use a generate-candidate-and-test approach that may generate a large amount of candidates for dense datasets. Many candidates do not appear in the database. Therefore we are introducing a more efficient algorithm for sequential pattern mining. The time complexity of proposed algorithm will be lesser in comparison to previous algorithms

I. Introduction

Data mining is the process of extracting interesting (non-trivial, implicit, previously unknown and potentially useful) information or patterns from large information repositories such as: relational database, data warehouses, XML repository, etc. Also data mining is known as one of the core processes of Knowledge Discovery in Database (KDD).

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 closelycorrelated sets but also their relationships in time are uncovered.

In a 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	$\{b\},\{f, g\}$

Table. 1: Sequential Data Base

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

A sequential rule typically has the form X->Y .A sequential rule X⇒Y has two properties:

- Support: the number of sequences where X occurs before Y, divided by the number of sequences.
- Confidence the number of sequences where X occurs before Y, divided by the number of sequences where X occurs.

Sequential Rule Mining finds all valid rules, rules with a support and confidence not less than user-defined thresholds *minSup*and *minConf*

Consider *minSup*= 0.5 and *minConf*= 0.5:

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

Table. 2: Sequential Data Base

ID	Rule	Support	Confidence
r1	$\{a, b, c\} \Rightarrow \{e\}$	0.5	1.0
r2	$\{a\} \rightarrow \{c, e, f\}$	0.5	0.66
r3	$\{a, b\} \rightarrow \{e, f\}$	0.5	1.0
r4	$\{b\} \rightarrow \{e, f\}$	0.75	0.75
r5	$\{a\} \rightarrow \{e, f\}$	0.75	1.0
r6	$\{c\} \rightarrow \{f\}$	0.5	1.0
r7	$\{a\} \rightarrow \{b\}$	0.5	0.66
…	…	…	…

Table. 3: Sequential Rules

II. A Survey Of Sequential Pattern Mining Methods

In general, we may categorize the mining approaches into the generate-and-test framework and the pattern-growth one, for sequence databases of horizontal layout. Typifying the former approaches [1,2 , 3], the GSP (Generalized Sequential Pattern) algorithm [3] generates potential patterns (called candidates), scans each data sequence in the database to compute the frequencies of candidates (called supports), and then identifies candidates having enough supports as sequential patterns. The sequential patterns in current database pass become seeds for generating candidates in the next pass. This generate-and-test process is repeated until no more new candidates are generated. When candidates cannot fit in memory in a batch, GSP re-scans the database to test the remaining candidates that have not been loaded into memory. Consequently, GSP scans at least k times of the on-disk database if the maximum size of the discovered patterns is k, which incurs high cost of disk reading. Despite that GSP was good at candidate pruning,

the number of candidates is still very huge that might impair the mining efficiency.

The PrefixSpan(Prefix-projected Sequential pattern mining) algorithm [4,12], representing the pattern-growth methodology [5, 4, 6], finds the frequent items after scanning the sequence database once. The database is then projected, according to the frequent items, into several smaller databases. Finally, the complete set of sequential patterns is found by recursively growing subsequence fragments in each projected database. Two optimizations for minimizing disk projections were described in [4]. The bi-level projection technique, dealing with huge databases, scans each data sequence twice in the (projected) database so that fewer and smaller projected databases are generated. The pseudo-projection technique, avoiding physical projections, maintains the sequence-postfix of each data sequence in a projection by a pointer-offset pair. However, according to [4], maximum mining performance can be achieved only when the database size is reduced to the size accommodable by the main memory by employing pseudo-projection after using bi-level optimization. Although PrefixSpansuccessfully discovered patterns employing the divide-and-conquer strategy, the cost of disk I/O might be high due to the creation and processing of the projected sub-databases.

Besides the horizontal layout, the sequence database can be transformed into a vertical format consisting of items' id-lists [7, 8, 9]. The id-list of an item is a list of (sequence-id, timestamp) pairs indicating the occurring timestamps of the item in that sequence. Searching in the lattice formed by id-list intersections, the SPADE (Sequential PAtternDiscovery using Equivalence classes) algorithm [9,10,11] completed the mining in three passes of database scanning. Nevertheless, additional computation time is required to transform a database of horizontal layout to vertical format, which also requires additional storage space several times larger than that of the original sequence database.

III. Proposed Algorithm

Step. 1 : Start
Step. 2 : Read the Sequence Database (DB) and MinimumSupport(MS)
Step. 3 : Find Frequent Item sets
Step. 4 : Arrange frequent items in lexicographic order & then assign a unique index to frequent itemsets
Step. 5 : Transform the DB into an equivalent data base and remove infrequent items from the data base..
Step. 6 : P= Ø (Set of all frequent sequential patterns)
Step. 7 : Create partitions of sequential patterns as P1 to Pn. This n is equal to the number of FI found. Then find prefix and suffix of each of P1 to Pn. At end P is formed by conquering P1 to Pn.
Step. 8 : Stop.
Example:

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

Table. 4: Sequential Data Base

For the above sequential data base the frequent itemsets are as follows:(1),(1,2),(2),(3). UniqueI disassignedto each FI as follows :- (1)-1, (1,2)-2, (2)-3, (3)-4[7]. The algorithmconsists oftwomajorsteps.

First the given sequential database is converted into an unique ID based data base. Infrequent items are also eliminated in this step.

Therefore, the converted sequential databaseis 1:<(1)(1,2,3,4)(4)>;2:<(4)(1,2,3)(3,4)>;3:<(1,2,3)(4)(1,2,3)>

The next task is problem partitioning. In this example, the number of frequent item sets are four. Therefore the final sequential pattern set will contain four disjoint subsets. These four disjoint subsets are P1, P2, P3, P4. Then we find prefix and suffix of each database.

First the calculation is performed for P4. The data base is converted in to DB4. The projected data base for P4. It is the data base for the fourth element i.e. 3.
It is as follows:
{1 : <(1)(1,2,3,4)(4)> ;2 :<(4),(1,2,3)(3,4)>;3 <(1,2,3)(4)(1,2,3)> }

Prefix of 4thDB is {1: < (1) (1,2,3,4) >; 2: < (4) (1,2,3) > ; 3: < (1,2,3) >} andSuffix of 4th DB is {1: <(4)> ;2: <(1,2,3) (3,4)> ;3: <(1,2,3)>} . It is clear that the suffix of this data base does not contain any frequent pattern.
prefix(3prefix (4D)) is 1: <(1)> ;2: <4>; sufixf (3prefix (4D)).
For prefix(2prefix (4D)) as 1:<(1)> ;2:<(4)> :
suffix (2prfixe (4D))
prefix(1prefix (4D)) 1: <(1)> ;2- <4> ;
suffix (1prefix (4D)) 1:<(1,2,3,4).
P4:<1 4><2 4><3 4><4>;
patterns for P3,P2,P1are found as
P3: <1 3><3>
P2:<2> ;
P1:<1> .
P is the union of all P1, P2, P3, P4. It is as follows
{<1 3> ,<1 4> ,<1> ,<2 4> ,<2> ,<3 4>, <3> ,<4>} .

IV. Time Complexity

If the length of longest pattern in sequence is N then the maximum number of projections that can be associated with that particular sequence will be N. From above it is clear that the every item in the sequence is scanned or checked a maximum of N times. In a database, if the total no of items in that data base is M and the average length of the sequential pattern is O. The total number of checks will be a maximum of OM. Therefore the time complexity of the proposed algorithm is O((log O)M) because the because the minimum levels of projections to detect a pattern with length N is about logN+1. In ODDAG, it is O(OM). because the minimal levels of projections to detect a pattern with length N is always N.

V. Conclusion

This paper presents a new and efficient algorithm to mine sequential patterns from a sequential data set. The time complexity of the proposed algorithm is lesser in comparison to the existing algorithms.

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.

[7] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal and M.-C. Hsu, "FreeSpan: Frequent Pattern-projected Sequential Pattern Mining," Proceedings of the 6th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 355-359, 2000.

[8] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal, "Multi-Dimensional Sequential Pattern Mining," Proceedings of the 10th International Conference on Information and Knowledge Management, pp. 81-88, 2001.

[9] J. Ayres, J. E. Gehrke, T. Yiu, and J. Flannick, "Sequential PAttern Mining Using Bitmaps," Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Edmonton, Alberta, Canada, July 2002.

[10] S. Parthasarathy, M. J. Zaki, M. Ogihara, and S. Dwarkadas, "Incremental and Interactive Sequence Mining," Proceedings of the 8th International Conference on Information and Knowledge Management, Kansas, Missouri, USA, pp. 251-258, Nov. 1999.

[11] M. J. Zaki, "SPADE: An Efficient Algorithm for Mining Frequent Sequences," Machine Learning Journal, Vol. 42, No. 1/2, pp. 31-60, 2001.

[12] Dr P padmaja, P Naga Jyoti, m Bhargava "Recursive Prefix Suffix Pattern Detection Approach for Mining Sequential Patterns" IJCA September 2011

