Novel Approach to Mine Sequential Frequent Pattern

Aashka Shah[1], Mr.Krunal Panchal[2]

[1] PG Student, Computer Engineering, LJIET, Ahmedabad, Gujarat, India
[2] Assistant Professor, Computer Engineering, LJIET, Ahmedabad, Gujarat, India

ABSTRACT

Sequential pattern mining is an important data Mining technique. Mining sequential rules from the sequence database is an important task with wide application. Its use to find frequently occurring ordered events or sub sequence as pattern from sequence database. Sequence can be called as order list of event. If one item set is completely subset of another item set is called sub sequence. Sequential pattern mining is used in various domains such as medical treatments, natural disasters, customer shopping sequences, DNA sequences and gene structures. The problem is to discover the all sequential pattern who satisfy the user specified constraint, from the given sequence database. There are various Sequential pattern mining algorithm like GSP, SPADE, SPAM, PrefixSpan are mainly used to find the relevant sequential frequent pattern from the sequence. All these sequential pattern mining algorithm are generating large set of frequent sequential pattern which are not time and memory efficient. CMRule, ERMiner, and RulrGrowth algorithms generate sequential rule but the method of generation is complicated, memory consumption is also high and it is not time efficient. So the Proposed novel approach is generating sequential frequent pattern as well as sequential rule in novel Method and it is more efficient in terms of memory and time.

Keyword : Sequential pattern mining, Sequence database, Sequential rule Mining.

1. INTRODUCTION

Sequential Pattern Mining finds interesting sequential patterns among the large database. It finds out frequent subsequences as patterns from a sequence database. With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining sequential patterns from their database. Sequential pattern mining is one of the most well-known methods and has broad applications including web-log analysis, customer purchase behavior analysis and medical record analysis. In the retailing business, sequential patterns can be mined from the transaction records of customers. . For example, having bought a notebook, a customer comes back to buy a PDA and a WLAN card next time. The retailer can use such information for analyzing the behavior of the customers, to understand their interests, to satisfy their demands, and above all, to predict their needs. Another example of sequential patterns is that in a book store's transaction database history, 80% customers who brought the book Database Management typically bought the book Data Warehouse and then brought the book Web Information System with certain time gap. All those books need not to be brought at the same time or consecutively, the most important thing is the order in which those books are brought and they are bought by the same customer. 80% here represents the percentage of customers who use this purchasing habit.

2. LITERATURE REVIEW

Sequential pattern mining is an important data mining problem, which detects frequent sequences in a sequence database. Sequential pattern mining is mainly classified into two parts, Apriori based and pattern growth based algorithm. The techniques for mine sequential pattern mining are described below.

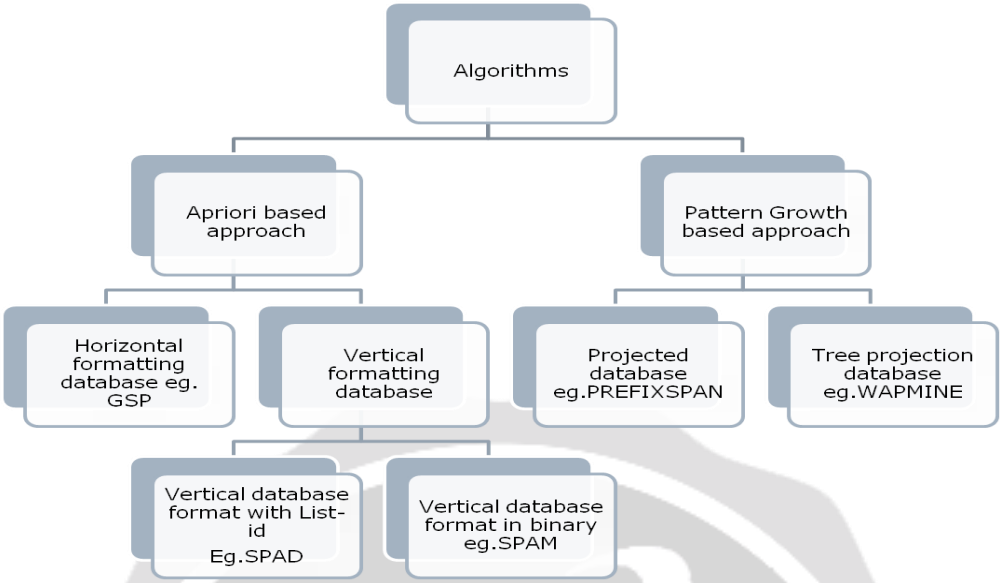

Fig.1 Taxonomy of Sequential Pattern Mining

2.1 Sequential pattern mining algorithm

2.1.1 Apriori based algorithm

1. GSP(Generalized Sequential Pattern):

GSP is apriori based algorithm[3]. It is much faster than Apriori all algorithm presented by Agrawal. It has a very good scale up properties with respect to the number of transactions per data sequence and number of items per transaction. But it is not efficient in mining large sequence of databases having numerous patterns.

2. SPAND(Sequential Pattern Discovery using Equivalence classes):

SPADE is an Apriori based vertical format sequential pattern mining. In additional this algorithm uses the ID-List techniques to reduce the cost for coming support counts. It consists of ID-List pairs where the first value stands for customer sequence and the second value reference to a transaction in it. Mining long sequence patterns using SPADE is not possible because of it needs exponential number of short candidates.

3. SPAM(Sequential Pattern Mining):

SPAM uses a vertical bitmap data structure representation of database which is similar to the given id-list of SPADE. It integrates the concept of GSP and SPADE algorithm. SPAM reduces the cost of merging but takes more time and space when compared to other algorithms which can be completely stored in the main memory.

2.1.2 Pattern Growth based algorithm

1. PrefixSpan (Prefix-projected sequential pattern mining):

PrefixSpan algorithm explores pattern growth approach for efficient mining of sequential pattern in large sequence database. These uses divide and conquer strategy with pattern growth approach in which sequential database is recursively projected into smaller projected database based on current sequential pattern. This algorithm requires smaller memory space than GSP and SPADE.

2. WAPMINE(Web access pattern mining):

Mining access pattern from weblog is called WAPMINE for that tree projection approach use with pattern growth approach. WAP tree store highly compressed critical information for access pattern mining and access the pattern in large set of log pieces [7]. This algorithm scan database twice which avoid problem of generating candidate set like apriori based. This algorithm is suffers from memory consumption problem because it construct recursively WAP tree when number of mined frequent pattern increase.

2.2 Sequential rule for frequent sequential Pattern

Sequential rules are discovering temporal relationships between events stored in large databases is important in many domains. It helps to understand the relationships between events and sets a basis for the prediction of events.

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.

1. CMRules Algorithm:

CMRules algorithm generates the sequential frequent pattern and derived sequential rules for those patterns. The algorithm calculates the sequential support and sequential confidence of each association rule by scanning the sequence database, and then it eliminates the rules that do not meet minimum thresholds. The set of rules that is kept is the set of all sequential rules. It relies on the property that any sequential rules must also be an association rule to prune the search space of sequential rule. It is much faster than other algorithm but rule generation process is complicated.

2. RuleGrowth Algorithm:

RuleGrowth algorithm use pattern growth mechanism for discover sequential rule generation. It relies on pattern growth algorithm so it is avoid candidate generation. RuleGrowth algorithm generates faster and batter sequential rules compare to CMRules. The drawback of this algorithm is it scans multiple time databases and it required more memory space.

3. ERMiner Algorithm:

ERMiner (Equivalence class based sequential Rule Miner) algorithm use a vertical representation of the database to avoid performing database projection and the novel idea of exploration the search space of rules using equivalence classes of rules having the same antecedent or consequent. Furthermore, it includes a data structure named SCM (Sparse Count Matrix) to prune the search space. It is faster than the all other algorithms but it consumes more memory.

3. PROBLEM STATEMENT

Sequential Pattern Mining Algorithms are mainly divided into two parts Apriori based approach and Pattern growth based approach which generate all sequential frequent patterns which require too much time to generate pattern and also require large amount of memory storage for all patterns. The sequential rule generation algorithms like CMRule , ERMiner and RuleGrowth algorithms generates sequential rules but they are complicated and consume more time as well as memory. These algorithms generate rules for whole sequence database. So it is better to generates rule from the frequent sequential pattern rather than the whole sequence database.

4. PROPOSED METHOD

Proposed system will generate sequential frequent pattern and rule which is generated by novel approach. The relationship between the pattern and the prediction of the occurrence of pattern can be identify. The proposed algorithm is more efficient in terms of time and memory consumption.

4.1 Algorithm step:

Input:
Sequence database (SDB), a threshold min_sup, a threshold min_conf.
Step 1:
Find frequent 1-itemset from Database
For i=0 to end of list
    If (key $_{item}$ is not in list)
        Add key $_{item}$ into list
    Else
        Key $_{item}$.List (T_{ij})
Step 1.1:
    If Key $_{item}$.List length < min_sup discard this items

Step 2:

Calculate for other K-itemset from frequent 1-itemset for using logical and operation

$$Key_{item1.item2} = Key_{item1}(T_{ij}) \cap Key_{item2}(T_{ij})$$

Step 3:

$Key_{itemi.itemi+1\ldots itemi+k}$ .Length < min_sup

Discard pattern

Step 4:

For each Sequence frequent Pattern make rule from it.

Suppose $\alpha \Rightarrow \beta$.

Find conf = $sup(\beta)/ sup(\alpha)$.

Step 5:

If (Conf >= min_conf)

Step 6:

Save this rule in output file

Output:

Frequent sequential rule

4.2 Theoretical Analysis:

Table 1: Transaction database

TID	Transactions
1	< {a,b} , {c}, {f}, {g}, {e} >
2	< {a,d}, {c}, {b}, {a,b,e,f} >
3	< {a}, {b}, {f}, {e} >
4	< {b}, {f,g,h} >

Create List(T_{ij}) of the transaction items.

In this example, Take (minimum support) Min_sup = 2 and (Confidence) Min_conf = 0.5 , remove less than min_sup items.

Table 2: Listing of items – List_ I(T_{ij})

TID (item)	Transaction List (T_{ij})	List(T_{ij}).length
$T_{(a)}$	$S_{1(1)}$, $S_{2(1)}$, $S_{2(4)}$, $S_{3(1)}$	4
$T_{(b)}$	$S_{1(1)}$, $S_{2(3)}$, $S_{2(4)}$, $S_{3(2)}$, $S_{4(1)}$	5
$T_{(c)}$	$S_{1(2)}$, $S_{2(2)}$	2
$T_{(d)}$	**$S_{2(1)}$**	**1**
$T_{(e)}$	$S_{1(5)}$, $S_{2(4)}$, $S_{3(4)}$	3

$T_{(f)}$	$S_{1(3)}$, $S_{2(4)}$, $S_{3(3)}$, $S_{4(2)}$	4
$T_{(g)}$	$S_{1(4)}$, $S_{4(2)}$	2

Table 3 : Listing of items – List_ II(T_{ij})

TID	**Transaction List (T_{ij})**	**List(T_{ij}).length**
T(a,b)	$S_{2(1,3)}$, $S_{3(1,2)}$	2
T(a,c)	$S_{1(1,2)}$, $S_{2(1,2)}$	2
T(a,e)	$S_{1(1,5)}$, $S_{2(1,4)}$,$S_{3(1,4)}$	3
T(a,f)	$S_{1(1,3)}$ $S_{2(1,4)}$ $S_{3(1,3)}$	3
T(a,g)	**$S_{1(1,4)}$**	**1**
T(b,e)	$S_{1(1,5)}$, $S_{2(3,4)}$, $S_{3(2,4)}$	3
T(b,f)	$S_{1(1,3)}$, $S_{2(3,4)}$ $S_{3(2,3)}$ $S_{4(1,2)}$	4
T(c,e)	$S_{1(2,5)}$, $S_{2(2,4)}$	2
T(c,f)	$S_{1(2,3)}$, $S_{2(2,4)}$	2
T(c,g)	$S_{1(2,4)}$, $S_{2(2,4)}$	2

Table 4 shows the final sequential frequent rules generate according to patterns.

Table 4 : Sequential frequent rule

Rules	**support**	**Confidence**
abc => e	0.5	1.0
a => cef	0.5	0.66
ab => ef	0.5	1.0
b => ef	0.75	0.75
a => ef	0.75	1.0
c => f	0.5	1.0
…	…	…

5. EXPERIMENTAL RESULTS

Eclipse is used for compiling and executing purpose. Eclipse is an integrated development environment (IDE) with using SPMF tools. It contains a base workspace and an extensible plug-in system for customizing the environment which mostly written in Java. Experiment was carried out on real-life datasets having varied characteristics. Those datasets are FIFA, SIGN and LEVIATHAN. In experiment the proposed system is compared with the existing algorithm. We consider two parameters execution time and memory required to justify our proposed algorithm.

Table 5 : The comparison of different approach

APPROACH	SIGN X = 0.4 Y = 0.75		LEVIATHAN X = 0.3 Y = 0.5	
	A (ms)	B (MB)	A (ms)	B (MB)
ERMiner algorithm	24233	184.6570	96245	685.2345
Proposed algorithm	20052	24.8389	89165	474.0468

Here, X = Minimum Support
Y = Minimum Confidence
A = execution time in mille seconds (ms)
B = Memory requirement in MB

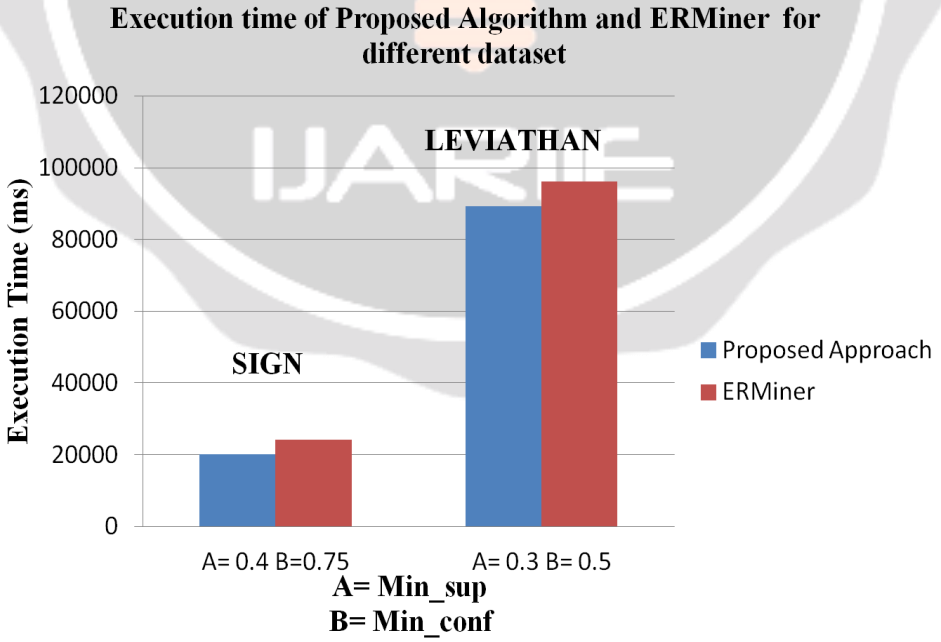

Chart 1: Execution Time comparison of ERMiner and Proposed Algorithm with different dataset

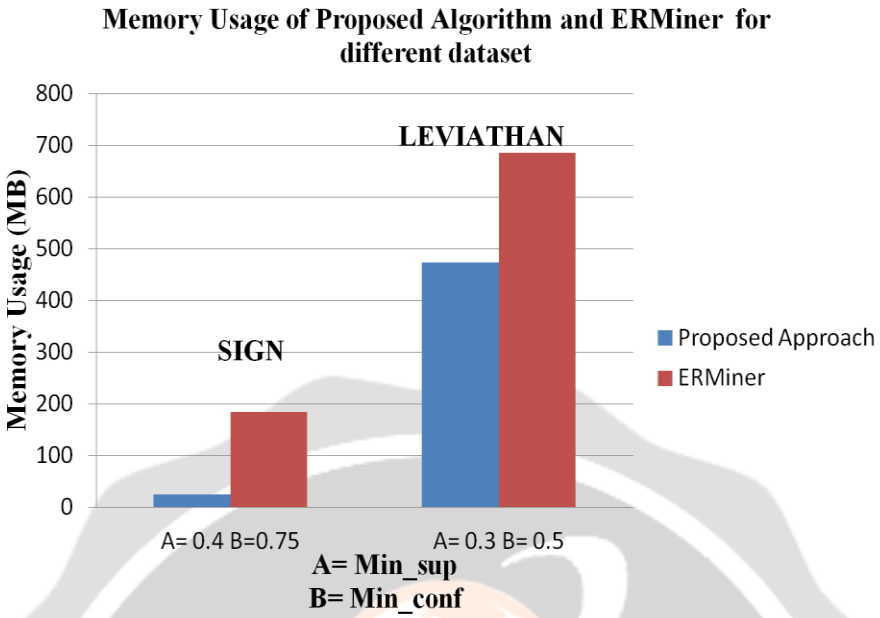

Chart 2: Memory Usage comparison of ERMiner and Proposed Algorithm with different dataset

6. CONCLUSIONS

The frequent sequential pattern mining and sequential rule mining are important task of data mining. According to observation sequential frequent pattern mining algorithm (like prefixspan,SPADE,SPAM) and closed sequential pattern mining algorithm (like Clospan, BIDE). Maximal sequential pattern mining algorithm gives very compact result compare to other algorithm. So, to store maximal sequential pattern require less memory storage and fast calculation compare two frequent sequential pattern and closed sequential pattern. Sequential rules generating algorithm like CMRules, Rulegrowth and ERMiner algorithm are complex, time and memory consuming. It is batter to generate rule from frequent pattern rather than whole sequential data sets.

7. REFERENCES

1. R. Agrawal, R. Srikant , "Mining sequential patterns," In Proceedings of Inter-national Conference on Data Engineering, pp. 3–14, 1995, ISBN:0-8186-6910-1.

2. Mabroukeh, N. R. and Ezeife, C. "A taxonomy of sequential pattern mining algorithms", ACM Computing Surveys, vol. 43, no. 1, pp. 1-41, 2010, DOI: 10.1145/1824795.1824798.

3. R. Srikant, R. Agrawal , "Mining sequential patterns: Generalizations and performance improvements," In Proceedings of International Conference on Extending Database Technology, pp. 3–17, 1996, DOI: 10.1007/BFb0014140.

4. Zaki, M. J. ,"SPADE: An efficient algorithm for mining frequent sequences", Machine learning, vol.42.no.1-2,pp.31-60,2001, DOI: 10.1023/A:1007652502315.

5. Ayres, J., Gehrke, J. Flannick, J., and Yiu, T., "Sequential Pattern mining using a bitmap representation", Proc. KDD 2002, Edmonton, Alberta, pp. 429-435, 2002, DOI: 10.1145/775047.775109.

6. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M., "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach", IEEE Trans. Knowledge and Data Engineering, vol. 16, no. 10, pp. 1-17, 2001, DOI: 10.1109/ICDE.2001.914830.

7. Fournier-Vinger, P., Nkambou, R., Tseng, V. S: "RuleGrowth: Mining Sequential rules common to several sequence by Pattern growth", Proc. of the 26th Symposium on applied computing, Taiwan, pp. 954-959, ACM Press,2011, ISBN: 978-1-4503-0113-8.

8. Fournier-Vinger, P., Usef Faghihi Nkambou, R., Engelbert Mephu, "CMRules: Mining Sequential rules common to several sequences", Elsevier 2012, DOI: 10.1016/j.knosys.2011.07.005.

9. Philippe Fournier-Vinger, Ted Gueniche, S. Zida and Vincent S. Tseng, "ERMiner: Sequential Rule Mining Using Equivalence Classes",pp 108-119,springer 2014.

BOOK:

10. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", second edition 2006, Morgan Kaufmann.

Web Links

11. http://help.eclipse.org/luna/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Fint_eclipse.htm, 26/02/2016,11:30 AM.