ISSN:0975-9646

Look Over on Mining Sequential Patterns in Evolving Data Stream

Harsha Nair [1], Neeba E A [2]

[1]PG Student, [2]Assistant Professor,
Department of Information Technology ,
Rajagiri School of Engineering and Technology,
Ernakulam, Kerala,India

***Abstract*— Data Stream are sequence of digitally encoded coherent signals (Packets of data or data packets) used to send or receive information that is in the process of being transmitted. It is a continuous, rapid and time-varying streams of data elements. A growing number of applications generate the streams of data. Such continuous generation of new elements in a data stream adds on additional constraints on the methods used for mining such data. Sequential pattern mining that performs mining in frequent patterns in time series database is a new area of data mining. Data evolution is one of the challenging problems of mining sequential patterns. In this paper, survey of some sequential pattern mining algorithms is done and comparison work has been carried out.**

***Keywords*— Streams, Sequential pattern mining, Sequential Patterns, Data evolution.**

I. INTRODUCTION

Data mining is one of the core processes of Knowledge Discovery in Database (KDD). As we know, data are changing every time, especially data on the web are highly dynamic. As time passes by, new data sets are inserted; old data sets are deleted data set, also it is important in the process of data while some other data sets are updated. It is obvious that time stamp is an important attribute of each mining and it can give us more accurate as well as useful information.

A database whose sequences of values or events change in accordance with time is called time-series database. For example, consider a time-series database that records the sales transaction of a supermarket in which each transaction includes an extra attribute that indicates when the transaction has happened. Time-series database is used to a great degree to store historical data in a diversity of fields such as medical
data, financial data, scientific data, etc. There are several different mining techniques designed for mining this data. Sequential pattern mining is for finding statistically relevant patterns between data samples where the values are delivered in any specific order. We can find the sequential patterns of specific individual item also we can find the sequential patterns among different items. Sequential pattern mining is widely applied in analyzing of DNA sequence and gene structures. It is the procedure of extracting certain sequential patterns whose support crosses a already defined minimal support threshold. It is often not possible to mine the sequential patterns with classical algorithms. Mainly because these algorithms require multiple scans over data stream, which is not feasible in a data stream environment. Hence, the mining algorithm should have only one pass over the incoming data records.

Various algorithms for sequential pattern mining in data streams were introduced, but these algorithms did not address the problem of evolving data with time. Hence, there is a need for a new sequential pattern mining algorithm that should be adopted to work in evolving data stream environments. It should be flexible, efficient, and uses simple data structures. In this work, we review the desired algorithm and compares with the previous algorithms. The paper is organized as follows: section 1 contains the introduction. Section 2 handles with the overview of concepts of data stream mining and windows on data streams. Section 3 details some of the algorithms of sequential pattern mining continued with Section 4 that compares the algorithms. Section 5 concludes the discussion.

II. DATA STREAMS

A data stream is a real-time, continuous and ordered sequence of items. It is impossible in data streams to control the order in which items arrive. Evolving data over time in data streams is one of the important factors that affect mining process. Data stream mining is the extracting of interesting patterns and trends from a sequence of elements that arrive continuously in a rapid rate. The continuity of data streams forces the mining algorithm to have only one pass or less over the incoming data records. Many times, users may not be interested in the contents of the whole stream since its beginning. But he may be interested in a portion of it defined by a window on the stream. Thus, windows on data streams and several methods to define such windows were introduced, which include fixed window, sliding window and landmark window. Here, we consider tilted-time and batch windows.

A. Tilted-Time and Batch Windows

The Tilted-time window is based on that the people are often interested in recent changes than in long term changes. The Fig. 1 shows such a tilted-time window like the most recent 4 quarters of an hour, then the last 24 hours and then 31 days. Based on this model, one can compute frequent item sets in the last hour with the precision of quarter of an hour, last day with the precision of an hour, and so on, until the whole month. This model registers only

4+24+31= 59 units. The maintenance of windows for natural tilted-time window is straightforward. After the four quarters are accumulated, they merge together to constitute an hour. After 24 hours are accumulated, a day is built.

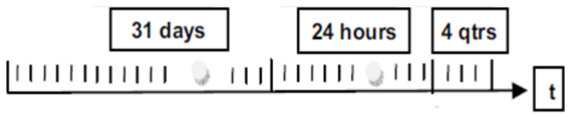

Fig. 1 Natural tilted-time window frames[1]

The authors in [13] introduced the logarithmic tilted-time window for storing patterns frequencies for recent changes and long-term changes. It is constructed based on a logarithmic time scale as shown in Fig. 2. Suppose the current window holds the transactions in the present quarter. If so, then the remaining slots are for the last quarter, next two quarters, 4 quarters, 8 quarters, 16 quarters, 32 quarters and so on growing at an exponential rate of 2. This schema is very space efficient.

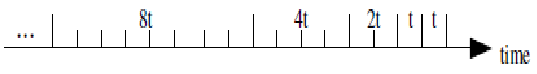

Fig. 2 Tilted-time window frames with logarithmic partition [1]

On the other hand, batch window model is a special case of sliding window model where there is no overlapping in consecutive windows. A batch Bi of stream sequential records consists of q sequential records S1, S2…Sq. For example, assume a window size of four records and a stream of twelve records S1, S2…S12. Hence, in batch window model, division of the records is as follows: the first batch window contains records S1, S2, S3 and S4, the second batch window contains records S5, S6. S7 and S8 and the third batch window contain records S9, S10, S11 and S12. The batch window model can be combined with tilted-time window model. It allows the user to obtain the set of sequential patterns over a period of time at any moment. it provides the flexibility to compute sequential patterns over user-defined time periods. It allows the user to choose between using natural tilted-time window model and logarithmic tilted-time window model. User can also specify the number of windows for mining and the window size. One of the sequential pattern mining algorithm, Prefixspan can be used for processing incoming batches one by one, as it is more efficient algorithm for mining sequential patterns in comparison with GSP and Apriori.

III. SEQUENTIAL PATTERN MINING ALGORITHMS

Apriori-like Algorithm series developed before prefixspan are: AprioriAll, GSP, SPADE and then the series of data projection based algorithms are Freespan and prefixspan.

A. *AprioriAll*

AprioriAll, the first algorithm for sequential pattern mining is based on the very first approach of Apriori association rule mining. AprioriAll was proposed to find the frequent sequential patterns. This is done in the sequence phase of data mining process. Similarly there are two sub-process. The first is to generate those sequences that may be frequent, which is also called candidate sequences. Second is that the sequential database is scanned to check the support of each candidate to determine the frequent sequential patterns according to minimal support. The time cost of the second process is determined by the number of passes over the database and number of candidates. Thus most researchers mainly take care
about the candidate generation process and passes over the database. The Apriori property is also used to prune those candidate sequences whose sub-sequence is not frequent. The main drawback of AprioriAll is that too many database passes are required and too much candidates are generated. It is not so efficient.

B. *GSP*

GSP is also an Apriori based algorithm for sequential pattern mining, but it doesn’t require finding all the frequent item sets first. This algorithm allows a) placing bounds on the time separation between adjacent elements in a pattern, b) allowing the items included in the pattern element to span a transaction set within a time window specified by user, c) permitting the pattern discovery in different level of a taxonomy defined by user. Additionally, GSP is designed for discovering generalized sequential patterns. The GSP algorithm makes multiple passes over sequence database as follows: 1) in the first pass, it finds the frequent sequences that have the minimum support. 2) At each pass, every data sequence is examined in order to update the occurrence number of the candidates contained in this sequence.
For the Apriori based algorithms, the number of candidate sequences is quite large and they require many passes over the whole database as well. But the two approaches are the basis of further researches on mining sequential pattern, at least all the sequential patterns can be generated though they are not so efficient. GSP performs relatively better than AprioriAll, in GSP the number of candidate sequences is much smaller, also time constraints, taxonomies are integrated during the sequential patterns mining process to produce more knowledge.

C. *SPADE*

SPADE is an algorithm proposed to find frequent sequences using efficient lattice search techniques and simple joins. All the sequences are discovered with only three passes over the database, it also decomposes the mining problem into smaller sub problems, which can be fitted in the main memory. SPADE outperforms AprioriAll and GSP by a factor of two through experiments. In this approach, the sequential database is transformed into a vertical id-list database format, in which each id is associated with corresponding items and the time stamp.

D. Freespan

FreeSpan is an algorithm with the aim to reduce the generation of candidate sub sequences. It uses projected databases to generate database annotations in order to guide the mining process to rapidly find frequent patterns. The general idea of FreeSpan is to use frequent items to project sequence databases into a set of smaller projected databases recursively using the currently mined frequent sets. Two alternatives of database projections can be used Level-by-level projection or Alternative-level projection. The method used by FreeSpan divide the data and the set of frequent patterns to be tested, and limits each test being conducted to the corresponding smaller projected database. FreeSpan scan the original database only three times, whatever the maximal length of the sequence. Experimental results show that FreeSpan is efficient and mines the complete set of patterns and it is considerably faster than the GSP algorithm. The major cost of FreeSpan is to deal with projected databases.

E. Prefixspan

Prefixspan is capable of dealing very large database. PrefixSpan mainly employs the method of database projection to make the database for next pass much smaller and consequently make the algorithm more speedy. Also in Prefixspan there is no need for candidates generation only recursively project the database according to their prefix. Different projection methods were discussed for Prefixspan: level-by-level projection, bi-level projection and pseudo projection.

The first step of Prefixspan is to scan the sequential database to get the length-1 sequence.After that the sequential database is divided into different partitions according the number of length-1 sequence, each partition is the projection of the sequential database that take the corresponding length-1 sequences as prefix. The projected databases only contain the postfix of these sequences. By scanning that database all the length-2 sequential patterns that have the parent length-1 sequential patterns as prefix can be generated. The projected database is then partitioned again by those length-2 sequential patterns. The same process is executed recursively until the projected database is empty or no more frequent length-k sequential patterns can be generated. The method mentioned above is called level-by-level projection, there is no candidate generation process. The cost mainly occurred in this method is the time and space used to construct projected databases.

Another projection method called bi-level projection is proposed to reduce the number and size of projected databases. The first step is the same, by scanning the sequential database we can get the frequent 1 sequence. In the second step a n x n triangle matrix M is constructed instead of constructing projected database. After that the S-matrix projected databases are constructed for those frequent length-2 sequences, all the processes iterated until the projected database becomes empty or no frequent sequence can be found. By using the triangle S-matrix to represent all supports of length-2 sequences, the number of projected databases becomes smaller and the requires less space.

Pseudo projection is another method designed to make the projection more efficient when the projected database can be fitted in main memory. Actually no physical projection database is constructed. Each postfix is represent by a pair of pointer and offset value. Pseudo projection is more efficient than the other two projection methods as it avoids copying the database, however the limitation is that the size of the database must be fitted into the main memory. Prefixspan mainly avoids generating and counting candidate sequences, which is the most time-consuming part of Apriori All and GSP. By using projection, the database Prefixspan scans every time is much smaller than the original database. The main cost of Prefixspan is the projected database generation process. In order to improve the performance a bi-level projection method that uses the triangle S-Matrix is introduced. Even if the sequential pattern is very long, Prefixspan can handle this more efficiently where GSP and Apriori All are unfeasible and inefficient. The main idea of Prefixspan algorithm is to use frequent prefixes to divide the search space and to project sequence databases. Its aim is to search the relevant sequences.

IV. COMPARISON

Depending on the management of the corresponding database, sequential pattern mining can be divided into three categories of databases, namely: a) Static database, b) Incremental database and c) Progressive database. Table 1 gives a comparison of the previously described algorithms based on the following features:

Database Multi-Scan: This feature includes the original database scanning to discover whether a long list of produced candidate sequences is frequent or not.

Candidate Sequence Pruning: This feature allows some algorithms (Pattern-growth algorithms, and later early-pruning algorithms) to utilize a data structure allowing them to prune candidate sequences early in the mining process.

DFS based approach: With the use of DFS search approach, all sub-arrangements on a path must be explored before moving to the next one.

BFS based approach: This feature allows level-by-level search to be conducted to find the complete set of patterns (All the children of a node are processed before moving to the next level).

Top-down search: This feature includes the following characteristic: the mining of sequential patterns subsets can be done by the corresponding set construction of projected databases and mining each recursively from top to bottom.

Bottom-up search: The Apriori-based approaches use a bottom-up search (from bottom to top), specifying every single frequent sequence.

Suffix growth vs Prefix growth: This feature allows that the frequent sub sequences exist by growing a frequent prefix/ suffix; since it is usually common among a good number of these sequences. This characteristic reduces the amount of memory required to store all the different candidate sequences sharing the same prefix/suffix.

Database vertical projection: This feature allows visiting the sequence database only once or twice to obtain a vertical layout of the database rather than the usual horizontal form, based on the bitmap or position indication table constructed for each frequent item.

TABLE I
COMPARATIVE STUDY OF SOME SEQUENTIAL PATTERN MINING ALGORITHMS

	AprioriAll	GSP	SPADE	Frees-pan	Prefixs-pan
Statistical Database	True	True	True	True	True
Database Multiscan	True	True			
Candidate Sequence Pruning		True	True		True
DFS Based Approach			True	True	True
BFS Based Approach		True			
Top-down Search				True	True
Bottom-up Search		True	True		
Prefix Growth					True
Database Vertical Projection			True		

V. CONCLUSIONS

Sequential pattern mining is indeed one of the wide topic in data mining.This mining technique is useful in various applications like in customer shopping sequence, natural disasters, DNA sequences, etc. In this paper, some of the sequential pattern mining algorithms were discussed along with their advantages and disadvantages. Additionally, a comparative analysis of some mining algorithms is done based on some features as defined in the previous section.

REFERENCES

[1] A. F. Soliman, G. A. Ebrahim and H. K. Mohammed, *SPEDS: A Framework for Mining Sequential Patterns in Evolving Data Streams,* IEEE, Ain Shams University, Cairo, Egypt, 2011.

[2] Q. Zhao and S. S. Bhowmick, *Sequential Pattern Mining: A Survey, Technical Report,* Nanyang Technological University, No. 2003118, Singapore, 2003.

[3] R. Agrawal and R. Srikant, *Mining Sequential Patterns,* In Proc. of the 11th Int. Conf. on Data Engineering, pp. 3-14, Taipei, Taiwan, 1995.

[4] R. Srikant, R. Agrawal, *Mining Quantitative Association Rules in Large Relational Table*, In Proc. of the ACM-SIGMOD 1996 Conference on Management of Data, Montreal, Canada, June 1996.

[5] M.J. Zaki, *Scalable algorithms for Association Mining,* IEEE Transactions on Knowledge and Data Engineering. 12(3), 372-390, 2000.

[6] J.Han , J.Pei, B. Mortazavi-Asl, Q. Chen, U.Dayal, And M.-C. Hsu, *Freespan: Frequent pattern projected sequential pattern mining,* In Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, 355359, 2000.

[7] R. Srikant and R. Agrawal, *Mining Sequential Patterns: Generalizations and Performance Improvements,* In Proc. of the 5th Int. Conf. on Extending Database Technology (EDBT96), pp. 3-17, Avignon, France, Mar. 1996.

[8] M. J. Zaki, *SPADE: An Efficient Algorithm for Mining Frequent Sequences*, Machine Learning Journal, Vol. 42, Issue 1/2, pp. 31-60, Kluwer Academic Publishers, Jan./Feb. 2001.

[9] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu, *Sequential Pattern Mining Using a Bitmap Representation*, In Proc. of the 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 429-435, Edmonton, Alberta, Canada, Jul. 2002.

[10] J. Pei, J. Han, B. Mortazavi-Asi, and H. Pinto, *PrefixSpan Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth*, In Proc. of the 17th Int. Conf. on Data Engineering (ICDE'01), pp. 215- 224, Heidelberg, Germany, Apr. 2001.

[11] L. Vinceslas, J.-E. Symphor, A. Mancheron, and P. Poncelet, *SPAMS: A Novel Incremental Approach for Sequential Pattern Mining in Data Streams, Advances in Knowledge Discovery and Management, Studies in Computational Intelligence,* Vol. 292, pp. 201-216, Springer, Berlin, Heidelberg, 2010.

[12] G. Hebrail, *Introduction to Data Stream Querying and Mining, Int. Workshop on Data Stream Management and Mining,* Beijing, China, Oct. 2008.

[13] C. Giannella, J. Han, J. Pei, X. Yan, and P.S. Yu. *Mining Frequent Patterns in Data Streams at Multiple Time Granularities,* In Proc. of the NSF Workshop on Next Generation Data Mining, pp. 191-212, Baltimore,USA,Nov.2002

.