Mining Sequential Patterns for interval based events by applying multiple constraints

M. Kalaivany and V. Uma

Department of Computer Science, Pondicherry University, Pondicherry

ABSTRACT

Sequential pattern mining finds the frequent subsequences or patterns from the given sequences. TPrefixSpan algorithm finds the relevant frequent patterns from the given sequential patterns formed using interval based events. In our proposed work, we add multiple constraints like item, length and aggregate to the interval based TPrefixSpan algorithm. By adding these constraints the efficiency and effectiveness of the algorithm improves. The proposed constraint based algorithm CTPrefixSpan has been applied to synthetic medical dataset. The algorithm can be applied for stock market analysis, DNA sequences analysis etc.

KEYWORDS

Sequential patterns, temporal patterns, Constraints, Interval based events.

1.INTRODUCTION

Data mining is useful in various domains such as market analysis, decision support, fraud detection, business management and so on. Sequential pattern mining is an approach to extract information from input sequences [1]. Various methods have been proposed for mining temporal patterns in sequence databases such as mining repetitive patterns, trends and sequential patterns. Sequential Pattern Mining is a popular technique which consists of finding subsequences appearing frequently in a set of sequence. However, knowing that a sequence appear frequently is not sufficient for making predictions. Sequential pattern mining approaches are classified as Apriori [2] or generate and test approach, pattern growth or divide-and-conquer approach. The Apriori and AprioriAll algorithms are based on apriori property and use the generate join procedure to form the candidate sequence. It identifies frequent item set in the database and extends it to a larger item set as those item set appears sufficiently in the database. Some of the widely used apriori based algorithms are GSP [3], SPADE [4] and SPAM [5].

Pattern growth algorithms allow the frequent item set discovery without candidate item set generation. They first build the data structure called FP-tree. Frequent Pattern tree consists of nodes corresponding to items and counters. This tree reads only one transaction at a time and maps it to a path. Then it extracts the frequent item set directly from the FP-tree. Some of the widely used pattern growth Algorithms are PrefixSpan [6] and FreeSpan [7]. A projection based pattern-growth method is used in PrefixSpan (Prefix-projected Sequential pattern mining) algorithm for mining sequential patterns. Its major idea is that, instead of projecting sequence databases by considering all the possible occurrences of frequent sub sequences, the projection is done on frequent prefix which results in higher efficiency of the algorithm in terms of processing time. The TPrefixSpan [8] algorithm is developed to mine the temporal patterns from interval-

DOI:10.5121/ijcsa.2014.4406

based events. Interval-based events are defined as the pair of time values associated with each event. Temporal pattern mining includes various tasks like temporal pattern analysis and comparison, temporal classification, temporal association rules, temporal clustering and temporal prediction [9]. TPrefixSpan algorithm deals with mining frequent temporal patterns from interval based events in a given sequence database. This paper discusses about adding multiple constraints with TPrefixSpan algorithm which increases the performance of TPrefixSpan algorithm and reduce the computational time. Constraints based PrefixSpan algorithm discovers sequential patterns which are frequent and also satisfy aggregate, length, and item constraints. In this work, Constraint based algorithm CTPrefixSpan is proposed to discover frequent temporal patterns considering item, length and aggregate constraints.

2. Literature survey

In [4] R.Srikant and R.Agarwal proposed an algorithm called "GSP" (Generalized Sequential Pattern) algorithm which is an apriori based approach for mining frequent patterns. SPAM: SPAM (Sequential Pattern Mining) [5] 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 [4], SPADE [6] and FREESPAN [7] algorithms. FreeSpan [7] is the frequent pattern projected sequential pattern mining algorithm which mines frequent sequences, finds the frequent patterns and uses the projected database to find the growth of subsequent fragments. Pie et al. [6] proposed a projection based sequential pattern mining algorithm called PrefixSpan algorithm (Prefix-projected Sequential Pattern mining), which mines the frequent patterns from a given data sequence. In [8] Wu-Chen proposed a non-ambiguous temporal pattern for interval-based events. TPrefixSpan algorithm was proposed to mine the new kind of temporal pattern from interval based events. Irfan Khan [10] included multiple constraints with the existing PrefixSpan algorithm to improve the efficiency of PrefixSpan algorithm. Similarly, to improve the efficiency of TPrefixSpan algorithm constraints are included and CTPrefixSpan algorithm is proposed in this work.

3. Existing system

Interval-based events are represented by two end points. The end point represent the start time and end time of the intervals. Projection method for interval-based events is different from that of point-based events. This form of representation is used in TPrefixSpan algorithm. TPrefixSpan algorithm is slower than PrefixSpan algorithm as representing the interval based events requires larger space.

Table 1. Example Sequence Database

Patient	Disease	Interval	Patient	Disease	Interval
01	a	{5, 10}	02	c	{16, 18}
01	b	{8, 12}	02	c	{19, 21}
01	c	{14, 18}	02	d	{16, 22}
01	d	{14, 20}	03	a	{12, 20}
01	b	{17, 22}	03	b	{22, 25}
02	a	{8, 14}	03	c	{29, 31}
02	b	{9, 15}	03	d	{28, 32}

In TPrefixSpan algorithm, initially the length of the sequential pattern is found. Then the set of frequent temporal 1-patterns is found from a given sequence database. These steps are repeated until the all frequent patterns are found. All frequent patterns from the sequences are generated. In TPrefixSpan algorithm "Temporal Prefix", and "Temporal Postfix" projected databases are to be found and the definitions are given below. The following 2 definitions are based on the definitions in [8].

3.1 Definition: (Temporal prefix)

Suppose we have two temporal sequences $\alpha = (p_1 op1\ p_2 op2 \ldots opn\text{-}1 p_n)$ and $\beta = (p'_1 op1'\ p'_2 op2' \ldots opm\text{-}1' p'_m)$, where $m \leq n$. Let p'_xdenote the last event starting point in β. Then, β is called a temporal prefix of α if and only if 1) β is contained in α,2) the events match, and 3) match in operators exist i.e.$op'_i = op_i$ (for $1 \leq i \leq x\text{-}1$).

3.2 Definition: (Temporal postfix)

Let α' be the projection of α with respect to β. Then temporal sequence γ is called the temporal postfix of α with respect to temporal prefix β and is denoted as $\gamma = \alpha / \beta$. If γ contains only one event starting point then we set $\gamma = \phi$.

4. Proposed system

To improve the efficiency of the existing TPrefixSpan algorithm, we need to add more constraints like length, item and aggregate on the output of frequent patterns. Multiple constraints based sequential pattern mining extracts the sequential patterns which are of user's interest.

By adding multiple constraints in interval based events the run time of finding frequent patterns from the given sequences will increases. We can use this constraints based algorithm in finding relevant sequences from the given medical data set and also to find the particular stock details in finance data set. It decreases the patterns from relevant frequent patterns according to the input sequences given. For any particular stock it gives the relevant patterns with respect to user's interest.

By adding multiple constraints in interval based events the run time of finding frequent patterns from the given sequences will increases. We can use this constraints based algorithm in finding relevant sequences from the given medical data set and also to find the particular stock details in finance data set. It decreases the patterns from relevant frequent patterns according to the input sequences given. For any particular stock it gives the relevant patterns with respect to user's interest.

Let us discuss how to generate patterns from given constraints by using our given algorithm.

4.1. Requirements:

4.1.1 To find the length and minimum support: From the given sequence database find the length and set minimum support initially as 0.5.

4.1.2 Apply TPrefixSpan algorithm: To mine the frequent patterns we are applying existing TPrefixSpan algorithm proposed by Wu and Chen [8]. TPrefixSpan algorithm is the pattern mining algorithm to find maximum number of frequent patterns from interval based events.

4.1.3 Constraints added: After applying TPrefixSpan algorithm the multiple constraints like item, length and aggregate constraints are added to refine the result according to user's interest. The various constraints added in our paper are explained below:

Item Constraints: An item constraint is defined as the subset of items that should or should not be present in the patterns [10]. For example,when mining sequential patterns over a stock market, a user may be interested in the patterns that have details about the day high price. The patterns that contain details about the day high are considered as interesting patterns. This enhances the effectiveness of the algorithm by providing patterns according to user's interest.

Length Constraints: A *length constraint* specifies the requirement on the length of the patterns, where the length can be either the number of occurrences of items or the number of transactions [10]. For example, a user may want to find longer patterns (i.e., at least 4 prices) in stock analysis. Such a requirement can be expressed by a length constraint and thus interesting patterns can be identified.

Aggregate Constraint: An *aggregate constraint* is the constraint on an aggregate of items in a pattern, where the aggregate function can be *sum*, *average*, *max*, *min*, *standard deviation* [10]. For example, a user may want to find the frequent sequential patterns where the average number of cost related to the prices is over 3.

By applying TPrefixSpan algorithm the frequent patterns mined are shown in Figure 2.

Frequent Sequence
HCL_start G HCL_end
INFOSYS_end G INFOSYS_start
IBM_start G IBM_end
HCL_start G HCL_end G INFOSYS_end G INFOSYS_start
IBM_start G IBM_end G HCL_start G HCL_end
IBM_start G IBM_end G INFOSYS_end G INFOSYS_start
IBM_start G IBM_end G HCL_start G HCL_end G INFOSYS_end G INFOSYS_start
INFOSYS_start G INFOSYS_end
IBM_end G IBM_start
HCL_start G HCL_end G INFOSYS_start G INFOSYS_end
IBM_end G IBM_start G HCL_start G HCL_end
IBM_end G IBM_start G INFOSYS_start G INFOSYS_end
IBM_end G IBM_start G HCL_start G HCL_end G INFOSYS_start G INFOSYS_end
HCL_end G HCL_start
HCL_end G HCL_start G INFOSYS_end G INFOSYS_start
IBM_end G IBM_start G HCL_end G HCL_start
IBM_end G IBM_start G INFOSYS_end G INFOSYS_start
IBM_end G IBM_start G HCL_end G HCL_start G INFOSYS_end G INFOSYS_start
IBM_end G IBM_start G HCL_start G HCL_end G INFOSYS_end G INFOSYS_start
HCL_end G HCL_start G INFOSYS_start G INFOSYS_end
IBM_start G IBM_end G HCL_end G HCL_start
IBM_start G IBM_end G INFOSYS_start G INFOSYS_end
IBM_start G IBM_end G HCL_end G HCL_start G INFOSYS_start G INFOSYS_end
IBM_start G IBM_end G HCL_end G HCL_start G INFOSYS_end G INFOSYS_start

Figure 2. Frequent patterns mined using TPrefixSpan algorithm

After applying Length constraint the resulting frequent patterns are shown in Figure 3.

Frequent Sequence
HCL_start G HCL_end G INFOSYS_end G INFOSYS_start
IBM_start G IBM_end G HCL_start G HCL_end
IBM_start G IBM_end G INFOSYS_end G INFOSYS_start
HCL_start G HCL_end G INFOSYS_start G INFOSYS_end
IBM_end G IBM_start G HCL_start G HCL_end
IBM_end G IBM_start G INFOSYS_start G INFOSYS_end
HCL_end G HCL_start G INFOSYS_end G INFOSYS_start
IBM_end G IBM_start G HCL_end G HCL_start
IBM_end G IBM_start G INFOSYS_end G INFOSYS_start
HCL_end G HCL_start G INFOSYS_start G INFOSYS_end
IBM_start G IBM_end G HCL_end G HCL_start
IBM_start G IBM_end G INFOSYS_start G INFOSYS_end
IBM_start G HCL_end G IBM_end G HCL_start
HCL_start G IBM_start G HCL_end G IBM_end
HCL_end G HCL_start G IBM_end G IBM_start
IBM_end G INFOSYS_end G IBM_start G INFOSYS_start
HCL_start G HCL_end G IBM_start G IBM_end
IBM_start G INFOSYS_end G INFOSYS_start G IBM_end
INFOSYS_end G INFOSYS_start G HCL_end G HCL_start
IBM_start G HCL_end G HCL_start G IBM_end

Figure 3. Frequent patterns after applying Length Constraint

After applying item constraint the resulting frequent patterns are shown in Figure 4.

Frequent Sequence
INFOSYS_end G INFOSYS_start
IBM_start G IBM_end
IBM_start G IBM_end G INFOSYS_end G INFOSYS_start
INFOSYS_start G INFOSYS_end
IBM_end G IBM_start
IBM_end G IBM_start G INFOSYS_start G INFOSYS_end
IBM_end G IBM_start G INFOSYS_end G INFOSYS_start
IBM_start G IBM_end G INFOSYS_start G INFOSYS_end
IBM_end G INFOSYS_end G IBM_start G INFOSYS_start
IBM_start G INFOSYS_end G INFOSYS_start G IBM_end
INFOSYS_end G INFOSYS_start G IBM_start G IBM_end

Figure 4. Frequent patterns mined after applying item constraints

The proposed CTPrefixSpan algorithm is given in Figure 5.

Algorithm: CTPrefixSpan ()

Input: Temporal Sequence Database S.
Output: Frequent patterns, Interesting patterns satisfying user specified constraints
Begin
Step 1: Identify frequent temporal 1-patterns in S.
Considering the frequent temporal 1-pattern generated identify the projected database.
Step 2: Generate frequent patterns with frequent 1-pattern as prefix.
Identify the projected database.
Repeat Step 1 and 2 until all the frequent temporal patterns are mined.
Get the constraint type from the user.
If constraint specified is item constraint **then**

Identify the patterns that has the item (event) specified.
Else if constraint specified is length constraint **then**
Identify the patterns with length= length specified
Else if constraint specified is aggregate constraint **then**
Identify the patterns that matches the aggregate specified
End if
Generate interesting patterns that satisfy the constraints
End

Figure 5. Pseudo-code for CTPrefixspan

. Result Analysis

Our CTPrefixSpan algorithm improves the efficiency and effectiveness than other sequential pattern mining algorithms. The runtime and Precision of both TPrefixSpan and CTPrefixSpan can be analysed by using medical or finance datasets. Though our algorithm increases the runtime, the precision rate of the proposed CTPrefixSpan algorithm is almost equal to 1.

5.1 Performance Evaluation

To evaluate the performance of our CTPrefixSpan, we implemented the TPrefixSpan algorithm for comparison. Both these algorithms were implemented in .NET language and tested on an Intel core windows 7 system using Micrososft Visual Studio package. The runtime of these two algorithms are compared with different length of output patterns. Finally, we calculated the Precision rate of TPrefixSpan and CTPrefixSpan algorithms with different data sets. Finally, it was found that our algorithm increases the runtime. But the precision rate is always nearest to value 1.

5.2 Runtime Comparisons

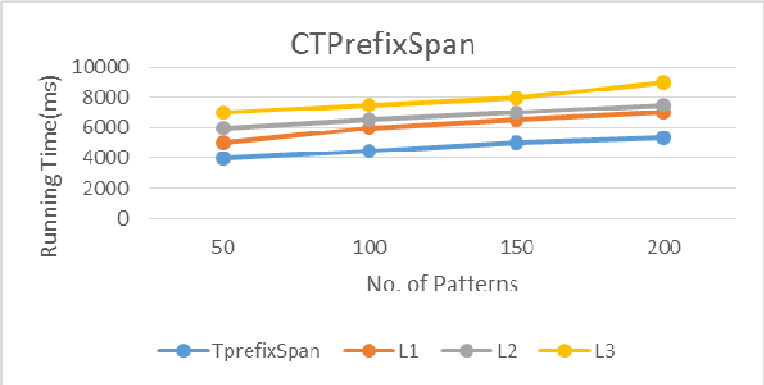

Figure 6. Comparison of RunTime

Figure 6 explains the run time with different number of patterns. The run time increases with different lengths like L1, L2, and L3 for CTPrefixSpan. But for existing TPrefixspan algorithm runtime is low than our CTPrefixSpan.

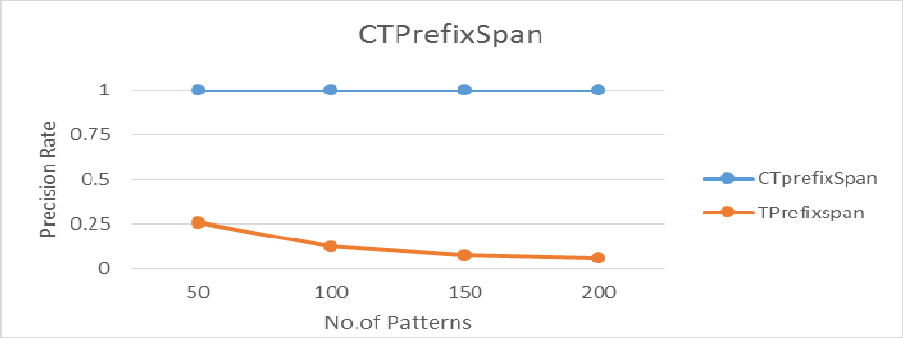

Figure 7. Precision Rate Comparison

$$\text{Precision} = \frac{\text{Relevant patterns} \cap \text{Retrieved patterns}}{\text{Retrieved patterns}}$$

The precision value is calculated by using relevant patterns and retrieved patterns with respect to TPrefixSpan and CTPrefixSpan algorithms. After calculating the precision it is found that our CTPrefixSpan algorithm has a precision rate of 1. Finally it can be concluded that after applying constraints with TPrefixSpan the runtime may increase but the precision rate of the proposed algorithm is high.

6. CONCLUSIONS

TPrefixSpan algorithm finds the relevant frequent patterns from the given data sequences representing interval-based events. It is the most efficient interval based algorithms among the sequential pattern mining algorithms. Adding multiple constraints like item and length to the mined patterns increases the scalability, accuracy and hence increases the efficiency of the proposed CTPrefixSpan algorithm.

REFERENCES

[1] R. Agrawal and R. Srikant, “Mining sequential patterns”, International Conference of Data Engineering (ICDE ’95), 1995.
[2] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules”, Proc. 1994 International Conference of Very Large Data Bases (VLDB ’94), pp. 487-499, Sept. 1994.
[3] R. Srikant and R. Agrawal, “Mining sequential patterns: Generalizations and performance improvements”, Proceedings of the 5th International Conference Extending Database Technology, 1057, pp. 3-17, 1996.
[4] M. Zaki, “SPADE: An efficient algorithm for mining frequent sequences”, Machine Learning, 2001.
[5] J.Ayres, J. Flannick, J. Gehrke, and T. Yiu, “Sequential pattern mining using a bitmap representation”, Proc.OfInternational Conference on Knowledge Discovery and Data Mining, 2002.
[6] J. Pei, J. Han, B. Mortazavi-Asi, H. Pino, "PrefixSpan: Mining SequentialPatterns Efficiently by Prefix- Projected Pattern Growth", ICDE'01, 2001.
[7] J. Han, G. Dong, B. Mortazavi-Asl, Q. Chen, U. Dayal and M.-C. Hsu,“FreeSpan: Frequent pattern-projected sequential pattern mining”, Proc. 2000 International Conference of Knowledge Discovery and Data Mining (KDD’00), pp. 355-359, 2000.
[8] Shin-Yi Wu and Yen-Liang Chen, “Mining Non-ambiguous Temporal Patterns for Interval-Based Events”, IEEE Transactions on Knowledge and Data Engineering, Vol.19, No.6, June 2007.
[9] S.Kalaivany, M. Gomathi, R.Sethukkarasi, “Mining Temporal Patterns for Interval-Based and Point-Based Events”, International Journal of Computational Engineering Research (IJCER), Vol 3, Issue 4, April 2013.

[10] Irfan Khan. "PrefixSpan Algorithm Based on Multiple Constraints for Mining Sequential Patterns", International Journal of Computer Science and Management Research, Vol. 1, Issue 5, December 2012.

Authors

M. Kalaivany is a student of M.Tech (Network and Internet Engineering) in the Department of Computer Science, School of Engineering and Technology, Pondicherry University, India. She was awarded the Anna University for B.Tech degree in Computer Science. She finished her M. Tech in the field of Network and Internet Engineering. She has published 1 research paper in international journals and in the proceedings of various international conferences.

V.Uma is working as Assistant Professor in the Department of Computer Science, School of Engineering and Technology, Pondicherry University, India. She was awarded the Pondicherry University gold medal for M.Tech degree in Distributed Computing Systems. She is pursuing her Ph.D in the field of Temporal knowledge representation. Her research interest includes Data mining, Semantic web. She has published 15 research papers in various international journals and in the proceedings of various international conferences.