Sequential Pattern Mining for Library System using Prefix Spanning

Ma Htaw Sein, Thin Zar Win
University of Computer Studies, KyineTong
nahtaw22@gmail.com ; thinzarwin07@gmail.com

Abstract

Sequential pattern mining is an important data mining problem with broad applications. Most of the sequential pattern mining methods, such as GSP (Generalized Sequential Pattern) and AprioriAll explore a candidate generation and test approach to reduce the number of candidates to be examined. These approaches may not be efficient in mining large sequence databases having numerous patterns and/or long patterns. The better algorithm for sequential pattern is based on pattern-growth, a divide-and-conquer algorithm that projects and partitions databases based on the currently identified frequent patterns and grow such patterns to longer ones using the projected databases. This paper presents mining sequential pattern from library database by prefixSpan algorithm, which explores prefix projection in sequential pattern mining. PrefixSpan mines the complete set of patterns but greatly reduces the efforts of candidate subsequence generation. Moreover, prefix-projection substantially reduces the size of projected databases and leads to efficient processing. The experimental results of our system are also discussed in this paper.

1. Introduction

The objective of sequential pattern mining is to find all frequent sequential patterns with a user-specified minimum support. The area of sequential pattern mining can be applied to many scientific and business domains. The proposed system intends to apply the PrefixSpan algorithm to mine the sequential patterns of borrowed books in the library. Suppose the transaction database contains list of students, list of borrowed books of each students, then "80% of the students borrow Basic Java Programming, then Advance Java and then J2EE" may be one of the valuable sequential patterns found.

This paper presents a framework for mining sequential patterns based on divide-and-conquer strategy of pattern growth called PrefixSpan algorithm. PrefixSpan algorithm divides the large sequential database into smaller projected databases. Different processes will be performed for each projected database, reducing the total processing time.

This paper is organized as follows. Section 1 is the introduction, section 2 is related work. Sequential pattern mining is presented in section 3. In section 4, sequential pattern by prefixSpan algorithm is described. Section 5 is the proposed system design and section 6 is the system implementation and sample case study for prefixSpan algorithm. Section 7 is the conclusion and future work of the system.

2. Related Work

There have been many studies to the efficient mining of sequential patterns or other frequent patterns in time-related data [2, 11, 9, 10, 3, 8, 5, 4]. Almost all of the previously proposed methods for mining sequential patterns and other time-related frequent patterns are Apriori-like, i.e., based on the Apriori property proposed in association mining [1], which states the fact that any super-pattern of a nonfrequent pattern cannot be frequent.

Based on this heuristic, a typical Apriori-like method such as GSP adopts a multiple-pass, candidate generation-and-test approach in sequential pattern mining. The first scan finds all of the frequent items which form the set of single item frequent sequences. Each subsequent pass starts with a seed set of sequential patterns, which is the set of sequential patterns found in the previous pass. The bottleneck of an Apriori-based sequential pattern mining method come from its step-wise candidate sequence generation and test.

Han and et.al [6] proposed FreeSpan that uses frequent items to recursively project sequence databases into a set of smaller projected databases and grows subsequence fragments in each projected database. FreeSpan mines the complete set of patterns and is efficient and runs considerably faster than the Apriori-based GSP algorithm. However, since a subsequence may be generated by any substring combination in a sequence, projection in FreeSpan has to keep the whole sequence in the original database without length reduction. Moreover, since the growth of a subsequence is

explored at any split point in a candidate sequence, it is costly.

This paper presents sequential pattern mining by PrefixSpan (i.e., Prefix-projected Sequential pattern mining). It examines only the prefix subsequences and projects only their corresponding postfix subsequences into projected database. In each projected database, sequential patterns are grown by exploring only local frequent patterns. PrefixSpan mines the complete set of patterns and is efficient and runs considerably faster than both Apriori-based GSP algorithm and FreeSpan.

3. Sequential Pattern Mining

Sequential pattern mining is to discover frequent transaction patterns such that the presence of a set of items is followed by another item in the time-stamp ordered transaction set. Let $\{i_1, i_2, i_3, \ldots, i_m\}$ be an itemset i where an itemset is a non-empty set of items and ik is an item. A sequence s is denoted as $<s_1, s_2, s_3, \ldots, s_n>$ where s is an order list of itemsets and s_j is an itemset whose items are purchased by a customer at the same transaction-time. For an item i_k, it can appear only once in s_j, but can appear multiple times in s_i and s_j with different transaction-time.

All the transactions of a customer, ordered by increasing transaction-time, is a customer-sequence. The support count for a customer-sequence is defined as the fraction of total customers who support this sequence. Each sequence satisfying a certain minimum support threshold (user-specified) is called a large sequence. Given a transaction database D and a minimum support threshold, we can define the problem of mining sequential patterns as finding the maximal large sequences among all the sequences with support count greater than or equal to. Each found maximal large sequence represents a sequential pattern. In addition, we will consider time constraints when finding sequential patterns, and this makes the found sequence patterns more useful.

4. Pattern Growth Method

Pattern-growth methods are a more recent approach to deal with sequential pattern mining problems. The key idea is to avoid the candidate generation step altogether, and to focus the search on a restricted portion of the initial database. PrefixSpan is the most promising of the pattern-growth methods and is based on recursively constructing the patterns, and simultaneously, restricting the search to projected databases. A projected database is the set of subsequences in the database, which are suffixes of the sequences that have prefix a. At each step, the algorithm looks for the frequent sequences with prefix a, in the corresponding projected database. In this way, the search space is reduced at each step, allowing for better performances in the presence of small support thresholds. In general, pattern growth methods can be seen as depth-first traversal algorithms, since they construct each pattern separately, in a recursive way.

The approach adopts a divide-and-conquer, pattern-growth principle as follows: Sequence databases are recursively projected into a set of smaller projected databases based on the current sequential pattern(s), and sequential patterns are grown in each projected databases by exploring only locally frequent fragments.

4.1 PrefixSpan Algorithm

PrefixSpan is a fast sequential pattern mining algorithm which extracts frequent sequences with depth-first search by executing sequence database projection operations recursively. PrefixSpan consists of sequence database projection operation, which is the most important process in the algorithm.

It is an approach of pattern growth method. Pattern growth is a method of frequent-pattern mining that does not require candidate generation. It is based on the FP-Growth algorithm for frequent pattern mining. PrefixSpan uses prefix projection to mine the complete set of frequent sequential patterns. The general idea of this approach is as follows:

- It follows the frequent single items, then compresses this information into a frequent-pattern tree or FP-tree.
- The FP-tree is used to generate a set of projected databases, each associated with one frequent item.
- Each of these databases is mined separately.

The algorithm builds prefix patterns, which it concentrates with suffix patterns to find frequent patterns, avoiding candidate generation. Prefix spanning algorithm extends the pattern-growth approach to instead mine sequential patterns. Figure 1 presents how prefixSpan is built for a sequence database.

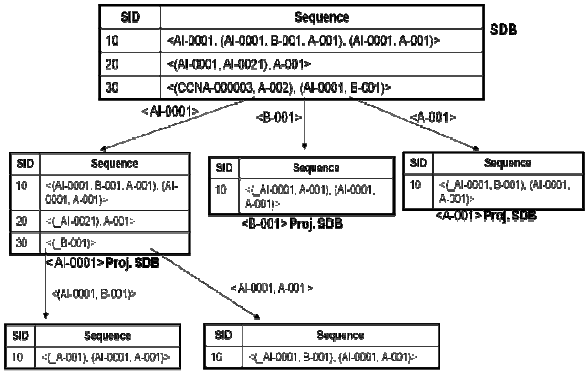

Figure 1: Building PrefixSpan

5. Proposed System

This paper presents mining sequential pattern from the sequence database of Library data using Prefix spanning algorithm. Students borrowing books at the library will be used as a transaction database. This transaction database is then transformed into sequence database. Then PrefixSpan algorithm with minimum support is then applied to that sequence database. Then frequent sequences are generated from the sequence database.

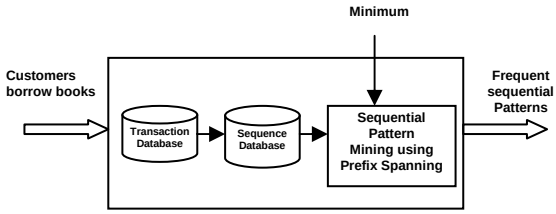

Figure 2: System overview

5.1. Process Flow of the System

Our system first reads the transaction database from online book store system. And then convert it into sequential database by grouping transactions of the same user based on transaction time. Then prefix items are computed and prune items whose support is less than minimum support. After that it prepares post-fix items according to item gap and time gap constraints. Finally, it counts sequences for generating frequent sequence patterns. Process flow of the system can be seen in Figure 3.

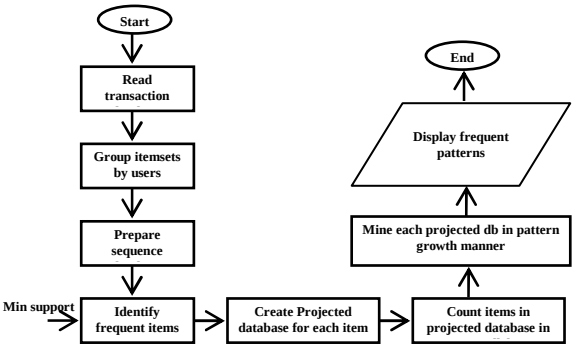

Figure 3: Process flow of the System

6. System Implementation

This system is an implementation of sequential pattern mining by PrefixSpan algorithm. Web-based library system is implemented to get the sequential database. Database system design is shown in Figure 4.

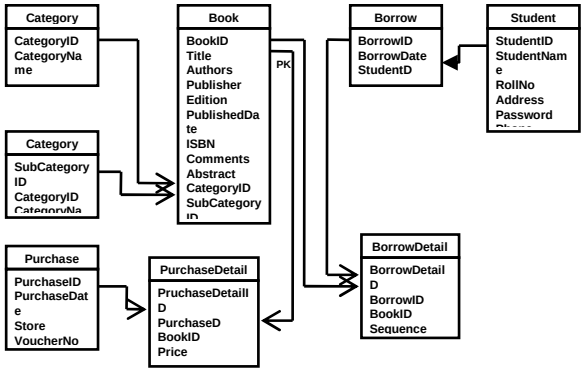

Figure 4: Database Design

6.1. Process of PrefixSpan Algorithm

Input: A sequence database S, and the minimum support threshold min_sup
Output: The complete set of sequential patterns
Method: Call PrefixSpan(<>, 0, S).
Subroutine PrefixSpan(a, l, S|a)
Parameters: a: a sequential pattern; l: the length of a; S|a: the a-projected database, if a != <>; otherwise, the sequence database S.
Method:
1. Scan S|a once, find the set of frequent items b such that
(a) b can be assembled to the last element of a to form a sequential pattern; or
(b) <b> can be appended to a to form a sequental pattern.
2. For each frequent item b, append it to a to form a sequential pattern a', and output a';
3. For each a', construct a'-projected database S|a', and call PrefixSpan (a', l+1, S|a')

6.2. Case Study

In this paper, borrow records of students are stored in the database and they are converted into sequential database of Library system as case study. Sample borrow List is shown in Table 1 and Table 2. If borrowID is same, our system groups the book list of same user to generate sequence based on time duration. Sequence database is shown in Table 3. Example sequential pattern extracted is shown in the following table 4.

Table 1: Borrow Header Table

BorrowID	BorrowDate	StudentID
1	1/2/2009	1
2	1/2/2009	2
3	1/28/2009	3

4	2/2/2009	3
5	4/6/2009	2

Table 2: Detailed Books for above borrow headers

BorrowID	BookID
1	PG-000005
1	PG-000006
2	PG-000005
2	PG-000006
3	PG-000005
3	PG-000006
4	AI-0001
4	AI-0003
5	AI-0003

Table 3: Sequential database

SID	Sequence
10	<PG-000005, PG-000006>
20	<PG-000005, PG-000006>, <AI-0003>
30	<PG-000005, PG-000006>, <AI-0001, AI-0003>

Table 4: Example Sequential Pattern

Sequence	Support
PG-000019, PG-000020	16
PG-000019, PG-000020, PG-000021	13
A-001, CCNA-000001	13
A-001, CCNA-000001, CCNA-000002	12
CCNA-000001, CCNA-000002, CCNA-000003, MCSE – 001	12

6.3. Experimental Results

This system is tested with 1000 transactions with 150 students. The main advantage of prefixSpan algorithm is the optimized processing time over Apriori-based algorithms. Figure 5 shows the performance analysis for processing times (in milli-seconds) of prefixSpan and Apriori-based algorithm.

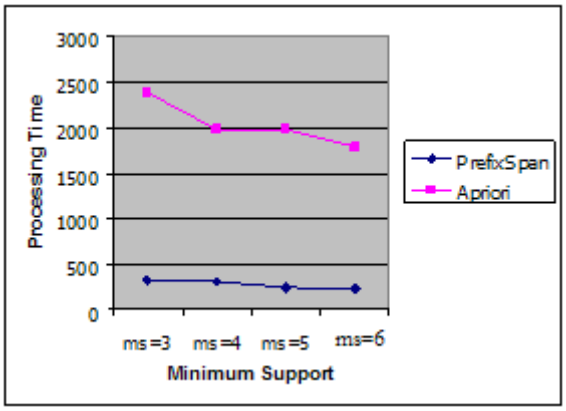

Figure 5: Processing Time (milli-seconds) of prefixSpan and Apriori-based algorithms.

The accuracy of the system is also tested with different minimum support and it is shown in Figure 6. Transaction database with transaction count = 1000 is used for all minimum support values.

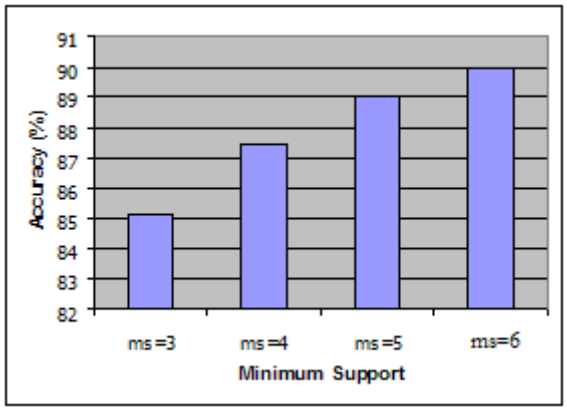

Figure 6: Accuracy of Our system

7. Conclusion

This paper presents the discovering of frequent sequential pattern from the transaction database. PrefixSpan algorithm is used to extract the frequent sequence patterns. It outperforms over the Apriori-based GSP algorithm because of divide-and-conquer approach, which avoids candidate generation and multiple database scan. It can be further implemented with the item intervals to get the more relevant frequent sequences, producing higher accuracy.

8. References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94), pages 487–499, Santiago, Chile, Sept. 1994.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In Proc. 1995 Int. Conf. Data Engineering (ICDE'95), pages 3–14, Taipei, Taiwan, Mar. 1995.

[3] C. Bettini, X. S. Wang, and S. Jajodia. Mining temporal relationships with multiple granularities in time sequences. Data Engineering Bulletin, 21:32–38, 1998.

[4] M. Garofalakis, R. Rastogi, and K. Shim. Spirit: Sequential pattern mining with regular expression constraints. In Proc. 1999 Int. Conf. Very Large Data Bases (VLDB'99), pages 223–234, Edinburgh, UK, Sept. 1999.

[5] J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. In Proc. 1999 Int. Conf. Data Engineering (ICDE'99), pages 106–115, Sydney, Australia, Apr. 1999.

[6] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu. Freespan: Frequent pattern-projected sequential pattern mining. In Proc. 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00), pages 355–359, Boston, MA, Aug. 2000.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00), pages 1–12, Dallas, TX, May 2000.

[8] H. Lu, J. Han, and L. Feng. Stock movement and ndimensional inter-transaction association rules. In Proc. 1998 SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery (DMKD'98), pages 12:1–12:7, Seattle, WA, June 1998.

[9] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1:259–289, 1997.

[10] B. O¨ zden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. In Proc. 1998 Int. Conf. Data Engineering (ICDE'98), pages 412–421, Orlando, FL, Feb. 1998.

[11] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In Proc. 5th Int. Conf. Extending Database Technology (EDBT'96), pages 3–17, Avignon, France, Mar. 1996.