Asian Journal of Information Technology 9 (4): 238-242, 2010
ISSN: 1682-3915

Discovery of Maximal Contiguous Sequence Patterns with Priority in Web Logs

[1]M. Thilagu and [2]R. Nadarajan
[1]VLB Janakiammal College of Engineering and Technology, Coimbatore, Tamil Nadu, India
[2]PSG College of Technology, Coimbatore, Tamil Nadu, India

Abstract: Frequent sequence mining has been a topic of research interest in recent years. Mining of web log sequence with forward references is an application of sequence mining and useful with many applications that include web recommendation, caching, pre-fetching etc. In this study, a fast algorithm presented that aims at mining prefix based maximal frequent contiguous sequences with priority in web logs efficiently. To enable fast mining, the algorithm first generates maximal potential patterns and discovers frequent patterns from them without candidate generation. Experimental results show that the proposed algorithm can perform better than existing approaches.

Key words: Contiguous sequence, sequence database, maximal contiguous sequences, algorithm, priority, maximal potential

INTRODUCTION

The expansion of the World Wide Web (WWW) has resulted in a large amount of data and retrieving the relevant information or hidden information from them is a challenging task. Web mining is the application of data mining techniques to extract knowledge from web data, i.e., web content, web structure and web usage data. Web mining can be broadly divided into three distinct categories according to the kinds of data to be mined and they are Web Content Mining is the process of extracting useful information from the contents of web documents, Web structure mining is the process of discovering the structure of hyperlinks in the web and Web usage mining is the task of discovering the activities of the users while they are browsing and navigating through the web.

Web usage mining also known as web log mining, aims to discover interesting and frequent user access patterns from web browsing data that are stored in web server logs, proxy server logs or browser logs. Web usage mining consists of three phases namely preprocessing, pattern discovery and pattern analysis. Preprocessing converts the original usage data into data abstractions necessary for patterndiscovery. The main preprocessing tasks include data cleaning, user identification, session identification and transaction identification. Pattern discovery adopts the various data mining techniques for discovering access patterns from the preprocessed usage data. The main techniques include statistical analysis, association rule mining, clustering, classification and sequential pattern mining. Once the access patterns have been identified they require to be analyzed to determine how that information can be used. Pattern analysis filters out uninteresting patterns from the set found during the pattern discovery phase. Web access patterns mined from web logs can be used for many practical applications include personalization, system improvement through pre-fetching, catching, site modification, e-business intelligence etc. (Nakagawa and Mobasher, 2003). Discovering access pattern is the problem of finding association rules in the web logs.

Mining frequent access patterns (called sequential access pattern mining) in a sequence database was firstly introduced by Srikant and Agrawal (1996). After that many studies have been contributed to the efficient mining of sequences and discussed on sequence mining (Antunes and Oliveira, 2003, 2004; Chen and Cook, 2007). Among the algorithms, the representative ones are GSP, FreeSpan, SPADE and PrefixSpan (Agrawal and Srikant, 1995; Han *et al*., 2000; Zaki, 2001; Pei *et al*., 2004) which consider the mining of generalized frequent sequences not satisfying contiguous property. The improved algorithms developed based on these representative methods are SPAM, MEMISP, LAPIN-SPAM and I-PrefixSpan (Ayres *et al*., 2002; Lin and Lee, 2005; Zhang and Kitsuregawa, 2005; Saputra *et al*., 2008). These algorithms studied on customer purchase behaviour sequences, however variants of these could be applied on web log sequences since a web access pattern is like a sequential pattern. Most existing

Corresponding Author: M. Thilagu, VLB Janakiammal College of Engineering and Technology, Coimbatore, Tamil Nadu, India

approaches for sequence pattern mining first detect short patterns and then gradually grow pattern length which may be inefficient due to large number of candidate sequence patterns. Several algorithms have been developed to mine patterns in web log sequences. Among them, the representative web access pattern techniques are Full Scan (FS) and Selective Scan (SS) proposed by Chen *et al.* (1998) used to mine traversal paths with maximal forward references and an algorithm was developed for mining frequent traversal sequences with backward references (Show-Jane, 2003). Pei *et al.* (2000) proposed an algorithm called WAP mine for finding access patterns from Web logs.

WAP mine constructs a WAP-tree and mines the sequences without candidate generation. However, the size of the structure for the WAP-tree is very large for long sequences and also needs to create a large number of links for linking to the next items.

A Contiguous Item Sequential Patterns (CISP) using a compact representation called UpDown Tree was proposed to mine contiguous frequent patterns in web logs. The algorithm generates prefix and suffix trees for each item by identifying the sequences containing that item and item based patterns are mined (Chen (2008).

In the study an algorithm without candidate generation has been proposed to mine frequent contiguous sequences with priority. It first generates prefix based maximal potential patterns and maximal frequent patterns are mined from them. Experiment results show that the approach outperforms better when compared to previous approaches.

PROBLEM STATEMENT

Given a web log database and the min-sup threshold, the problem defined here is to mine a complete set of prefix based maximal contiguous sequence patterns with priority in the database with a motivation that link between web pages is important and to be maintained in web log sequences.

The constraint in the problem is that sequences are with single occurrence of events without repetition. Examples are forward traversal path sequences without backward reference of web pages and sequences in sessions without repetition of pages in a web log.

Example 1: Consider a sequence database as shown in Table 1 and min-sup = 2. From the Table 1, the maximal frequent contiguous sequences prefixed with could identified item a that would satisfy the given min-sup as {abef, afdbe}.

Table 1: Sequence database

Seq-id	Sequence
1	a b e f c d
2	a f d b e c
3	a b e c d f
4	c a f d b e
5	c a b e f d

MINING OF MAXIMAL CONTIGUOUS SEQUENCE PATTERNS WITH PRIORTY

The proposed algorithm for mining maximal contiguous sequence patterns with priority is described. It first reads all the sequences from the database and finds the frequent 1-itemset. Initially, the database is projected with frequent 1-itemset α and frequent 2-itemset prefixed with α is found. The database is again projected for each frequent 2-itemset β as a prefix and maximal potential patterns are generated in lexicographical order by merging frequent patterns with 2-itemset detected from the projected sequences. Then, maximal frequent patterns are mined by scanning only those maximal potential patterns in the projected database using selective scan. Finally, the discovered patterns are assigned with priorities based on the confidence of suffix patterns in them. The above said process is recursively done for each frequent 1-itemset in the database.

PRELIMINARIES

Definition 1 (Contiguous sequence pattern): A contiguous sequence pattern is a frequent pattern in which items are adjacent or satisfy contiguous property in the underlying pattern with support count greater than or equal to min-sup.

Definition 2 (Maximal contiguous sequence pattern): A contiguous sequence pattern is said to be maximal if it is not a subsequence of any other frequent pattern.

Example 2: Consider the projected sequence database of item a as shown in Table 1 with min-sup = 2. The algorithm first discovers the frequent contiguous 2-itemset as {ab, af} in the projected database. For each frequent 2-itemset, the projected database is scanned to find the frequent patterns with 2-itemset. For example, frequent contiguous patterns with 2-itemset {be, ef, cd} are found from the sequences prefixed with ab. Then, these patterns are merged to generate a maximal potential pattern abef and we find the support count of the merged pattern abef using selective scan with a transaction-id list {1, 3, 5} in which the contiguous pattern ab exists. Then, these transactions are scanned to check whether the items in abef are in order and satisfy contiguous property. Since it exists in {1, 5} satisfying the given min-sup, it is a

maximal frequent pattern. Similarly, the pattern afdbe prefixed with af is generated by doing the above said process recursively and it is found to be a maximal pattern. Like this, maximal potential patterns are generated for remaining 2-itemsets and mined.

Algorithm
Input: Sequence database D, min-sup
Output: Maximal patterns with Priority

Step 1: Read the database D and find all the frequent 1-itemset: Repeat the steps in Phase-I and Phase-II for each frequent 1-itemset α to mine all maximal patterns in the database. During Phase-I, pruning process is done to check whether the generated pattern already exists or not.

Phase-I
Step 2: Generating prefix based maximal potential patterns: Scan the projected database and find the frequent 2-itemset prefixed with α Satisfying contiguous property. For each frequent 2-itemset β in the projected database:

- Scan the projected sequences with prefix β
- Find the frequent patterns with 2-itemset
- Generate prefix based maximal potential patterns by merging those
- Frequent patterns and do steps in Phase-II

Phase-II
Step 3: Mining maximal patterns:
Step 1: Arrange the prefix based maximal potential patterns in ascending order to classify the patterns with common prefix pattern.

Step 2: If a class Ci is having a single pattern Pi
Find the support count of pattern Pi by scanning the projected database once
Else
Find a transaction-id list T of common prefix pattern of Ci by scanning the projected database once
If the common prefix pattern is frequent
For each pattern Pi that belongs to Ci
Find the support count of suffix patterns of pattern Pi using the transaction-id list T
If support count>=min-sup then
Pi is a maximal frequent pattern
Endif
Endfor
Endif
Endif

Here step 1, 2 is needed to minimize the search space by finding the transactions in which the common prefix exists. For example, consider the maximal potential patterns say abcde, abced and abcfe with common prefix abc, then the transaction-id list of abc is first found and it is used for mining suffix patterns (de, ed, fe) with common prefix as abc, iff abc is frequent.

Priority of patterns: The priority of a maximal frequent pattern P is determined based on the parameters such as the support counts of suffix patterns, maximum support count of a suffix pattern and the length of the pattern P. Hence, the priority of a pattern P is defined as:

$$\text{Priority (P)} = \frac{\text{Sum of support counts of suffix patterns of P}}{\text{(Maximum support count of a suffix pattern x Length of pattern P)}}$$

Here the range of priority lies between 0.0 and 1.0, since a scaling factor i.e., maximum support count of a suffix pattern is used to normalize the priority value. For example, priority of patterns abef and afdbe are computed as 13/20 (0.65) and 13/25 (0.52) by applying the above formula. Like this, priorities of other patterns are found and the values are ordered in descending order based on pattern length.

As discussed earlier, the proposed algorithm mines maximal patterns with priority fast in two phases. That is it first generates the maximal potential patterns using projected sequences with frequent patterns. Then, it mines only those generated patterns for discovering frequent patterns using selective scan. To conclude, the merits of the proposed algorithm are No candidate generation, Minimized number of database scans due to maximal potential patterns which lead to fast mining of patterns.

PERFORMANCE EVALUATION

The experimental results on the performance of the study is reported in comparison with PrefixSpan, an efficient SP mining algorithm among several algorithms (Pei *et al.*, 2004). In the study, a variant of PrefixSpan algorithm has been applied on web log sequences and compared with the approach. However, the PrefixSpan algorithm suffers a lot from the construction of projected databases and multiple scans of the projected database. In worst case for each sequential pattern, the database is scanned in PrefixSpan algorithm. The algorithm overcomes this problem by generating maximal potential patterns to avoid level by level candidate generation and scanning of those generated candidate sequences. Hence it requires only 3 scans for a sequential pattern in the

worst case. The memory space required for this algorithm mainly holds prefix based frequent 2-itemset candidate sequences, maximal potential patterns and sequence datasets.

The machine used for performance evaluation was an IBM compatible personal computer with 2GHz Pentium microprocessor, 2GB RAM and the Windows XP operating system. The dataset used for performance evaluation is the CTI Dataset containing the preprocessed and filtered sessionized data for the main DePaul CTI Web server (http://www.cs.depaul.edu). Here, the file used for the experiment is the cti.tra file which contains the filtered sessionized data in transaction format. Each line in this file corresponds to the sequence of pages visited during one session. While the order of occurrence of pageview in each session represents the order in which these pageviews were visited, the transactions do not contain repeated visited to the same pageview in the same session.

Thus, only the first access to a pageview is recorded as part of the transaction. The performance evaluation of the algorithms is evaluated based on their time complexity and it is shown as a comparative study on the execution time of both the algorithms for the min-sup = 0.01 in Table 2. Figure 1 shows the performance analysis of these two algorithms on CTI dataset and a graph is drawn by using the execution time details of both the algorithms

Table 2: A comparative study on execution time in seconds

No. of sequences	Proposed algorithm	PrefixSpan
1K	18	25
2K	56	70
3K	107	140
4K	194	258
5K	366	539

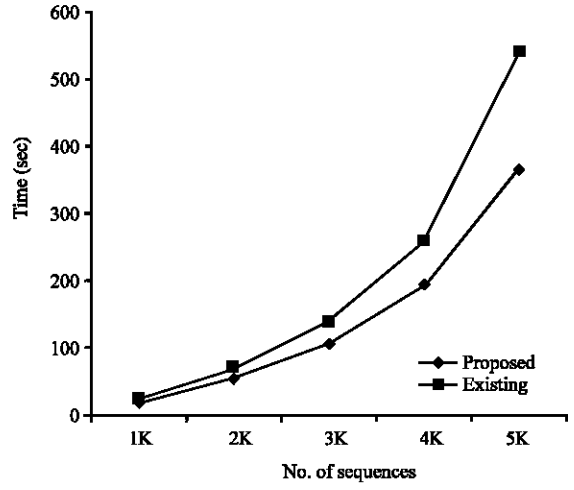

Fig. 1: Performance analysis for min-sup = 0.01

shown in Table 2. That is, the analysis is performed by varying the data sizes with the support count as 0.01. Experimental studies have been done having different the data sizes and the support counts. From the performance analysis report, it is found that the proposed algorithm performs better than PrefixSpan.

CONCLUSION

In this study we have explored a fast method for mining prefix based maximal contiguous sequence patterns with priority in a database. To enable fast mining, maximal potential patterns are generated and mining is performed only for those patterns. The proposed algorithm could improve the mining performance significantly in order to mine maximal patterns by avoiding candidate generation and reducing the search space on the database. Experimental results and evaluations performed on real data demonstrate that the proposed algorithm is very effective and outperforms existing algorithms.

REFERENCES

Agrawal, R. and R. Srikant, 1995. Mining sequential patterns. Proceedings of the 11th International Conference on Data Engineering, March 6-10, Taipei, Taiwan, pp: 3-14.

Antunes, C. and A.L. Oliveira, 2003. Generalization of pattern-growth methods for sequential pattern mining with gap constraints. Machine Learning Data Mining, 2734: 239-251.

Antunes, C. and A.L. Oliveira, 2004. Sequential pattern mining algorithms: Trade-offs between speed and memory. Proceedings of the 2nd Workshop on Mining Graphs, Trees and Seq, Italy.

Ayres, J., J. Gehrke, T. Yu and J. Flannick, 2002. Sequential pattern mining using a bitmap representation. Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, ACM Press, New York, USA., pp: 429-435.

Chen, J. and T. Cook, 2007. Using d-gap patterns for index compression. Proceedings of the 16th International Conference on World Wide Web, Banff, Alberta, Canada, May 08-12, ACM, New York, USA., pp: 1209-1210.

Chen, J., 2008. Contiguous item sequential pattern mining using updown tree. Intell. Data Anal., 12: 25-49.

Chen, M.S., J.S. Park and P.S. Yu, 1998. Efficient data mining for path traversal patterns. IEEE. Trans. Knowledge Data Eng., 10: 209-220.

Han, J., J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal and M. Hsu, 2000. FreeSpan: Frequent pattern-projected sequential pattern mining. Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 20-23, Boston, Massachusetts, USA., pp: 355-359.

Lin, M. and S. Lee, 2005. Fast discovery of sequential patterns through memory indexing and database partitioning. J. Inform. Sci. Eng., 21: 109-128.

Nakagawa, M. and B. Mobasher, 2003. A Hybrid Web Personalization Model Based on Site Connectivity. Proceedings of the In WEBKDD, pp: 59-70

Pei, J., J. Han, B. Mortazavi-Asl, J. Wang and H. Pinto *et al*., 2004. Mining sequential patterns by pattern-growth: The prefixSpan approach. IEEE. TKDE., 16: 1424-1440.

Pei, J., J. Han, B. Mortazavi-asi and H. Zhu, 2000. Mining Access Patterns Efficiently from Web Logs: In Proceedings of 6th Pacific Area Conference on Knowledge Discovery and Data Mining (PAKDD). Lecture Notes Comp. Sci., 1805: 396-407.

Saputra, D., D.R.A. Rambli and O.M. Foong, 2008. Mining Sequential Patterns Using I-PrefixSpan. Int. J. Comput. Sci. Eng., 2: 2-2.

Show-Jane, Y., 2003. An Efficient Approach for Analyzing User Behaviors in a Web-Based Training Environment. J. Dis. Edu. Techno., 1: 55-71.

Srikant, R. and R. Agrawal, 1996. Mining sequential patterns: Generalizations and performance improvements. Proc. Int. Conf. Extend. Database Technol., 1057: 3-17.

Zaki, M.J., 2001. SPADE: An efficient algorithm for mining frequent sequences. Mach. Learn. J., 40: 31-60.

Zhang, Z. and M. Kitsuregawa, 2005. LAPIN-SPAM: An Improved Algorithm for Mining Sequential Pattern. Proceedings of the International Special Workshop on Databases for Next Generation Researchers, April 5-8, University of Tokyo, pp: 1222-1222.