2012 4th International Conference on Computer Modeling and Simulation (ICCMS 2012)
IPCSIT vol.22 (2012) © (2012) IACSIT Press, Singapore

Pattern Finder–Efficient Framework for Sequential Pattern Mining

Mahbubul Arefin Khan [1], Md. Shafiul Alam Khan [1], Asfara Rehmina Towfiq[2], Faisal Ahmed Manzur[2]

[1]University of Dhaka Dhaka, Bangladesh
mk.arefin@gmail.com, shafiul@univdhaka.edu
[2] American International University, North South University Dhaka, Bangladesh
{towfiq.asfara, faisal.m.ahmed}@gmail.com

Abstract. With the proliferation of Internet, discovery and analysis of useful information from the World Wide Web becomes a practical necessity. Web usage mining has become a fertile field of research for improving designs of web sites, analyzing system performance as well as network Communications, understanding user reaction, motivation and Building adaptive Web sites. An important application of sequential mining techniques is web usage mining, for mining web log accesses, where the sequences of web page accesses made by different web users over a period of time, through a server, are recorded. We propose a framework of sequential patter mining- Pattern Finder which uses a proposed algorithm r-WAP. WAP-tree algorithm then, mines the frequent sequences from the WAP-tree by recursively re-constructing intermediate trees, starting with suffix sequences and ending with prefix sequences. The proposed algorithm r-WAP can find access patterns from Web logs quite efficiently. We have also added data preprocessing method that enable to handle large volume of raw data to make the state space of r-WAP tractable. The experimental and performance studies show that the proposed algorithm is in general an order of magnitude faster than existing algorithms.

Keywords: Data Mining, Web Mining, Web Access Pattern Tree

1. Introduction

Web mining is the application of data mining techniques to discover patterns from the Web [1]. In Web Mining, data can be collected at the server-side, client-side, proxy servers, or obtained from an organization's database (which contains business data or consolidated Web data). The information gathered through Web mining is evaluated (sometimes with the aid of software graphing applications) by using traditional data mining parameters such as clustering and classification, association, and examination of sequential patterns [2]. According to analysis targets, web mining can be divided into three different types, which are Web usage mining, Web content mining and Web structure mining. Web Usage mining has a lot of application in real life such as Improving designs of web sites, analyzing system performance as well as network Communications, understanding user reaction, motivation and Building adaptive Web sites; it is now a very important and useful subject. A novel data structure, called Web access pattern tree or WAP-tree in short, is developed for efficient mining of access patterns from pieces of logs. The Web access pattern tree stores highly compressed, critical information for access pattern mining and facilitates the development of novel algorithms for mining access patterns in large set of log pieces. WAP-tree algorithms do its work with two passes. One pass for finding frequent pattern mining and another for constructing the tree; which is time consuming. Our main target is to design an efficient algorithm using WAP-tree by reducing the total CPU time. For this we do the work in one pass.

2. Proposed Web Access Sequential Pattern Mining Algorithm

The WAP-mine algorithm mines sequential patterns by constructing WAP tree from sequence database. It developed for mining the complete Web access patterns from large set of pieces of Web log. We proposed an efficient algorithm works by scanning the whole database once. It builds the WAP-tree while scanning the database. That means time elapsed for finding frequent event is dropped. Mining procedure of r-WAP is same as the existing algorithm except it does not create conditional sequence base as well as conditional WAP-trees for all events in the header table. Count variable in the header table helps to do this.

Algorithm r-WAP: Mining access patterns in Web access sequence database.

Input: access sequence database WAS and support threshold € (0 < € < 1).

Output: the complete set of €-patterns in WAS.

1. Scan WAS, construct a WAP-tree over the set of individual event.

2. Recursively mine the WAP-tree using conditional search.

In this subsection, an illustration of how r-WAP algorithm works is stated using the same example used to illustrate existing algorithm.

Tab 1.0 A database of Web access sequences

User ID	Web Access Sequence
100	Abdac
200	Abcac
300	Babac
400	Aback

In Table 1.0 same database of Web Access Sequence is stated. Support threshold is close same that is 30%. Using proposed algorithm to construct the WAP tree, a virtual root (Root) is first inserted. Then, each sequence in the transaction is used to construct a branch from the Root to a leaf node of the tree. Each event in a sequence is inserted as a node with count 1 from Root if that node type does not yet exist, but the count of the node is increased by 1 if the node type already exists. Also, the head link for the inserted event is connected to the newly inserted node from the last node of its type that was inserted or from the header node of its type if it is the very first node of that event type inserted. A count of individual event is updated whenever the corresponding event is found in a sequence. In this way this header count corresponds to the total occurrence of the event in the database

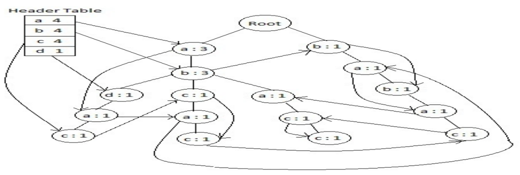


Fig.1 Complete WAP tree for a web access sequence database

Once the sequential data is stored on the complete WAP-tree (figure 4.1), the tree is mined for frequent patterns starting with the lowest frequent event in the header list. Checking for an event is frequent or not is done now with the help of header count. To qualify as a frequent event, one event must have a count of 2. In our example, d has the lowest frequency count but it is not meet the support threshold. So, we start from frequent event c as the following discussion shows. From the WAP-tree of figure 4.1, it first computes prefix sequence of the base c or the conditional sequence base of c as:

abda:1; abca:1; abac:1; baba:1.

The conditional sequence list of a suffix event is obtained by following the header link of the event and reading the path from the root to each node (excluding the node). The count for each conditional base path is the same as the count on the suffix node itself. The first sequence in the list above, abda represents the path to the first c node in the WAP tree.

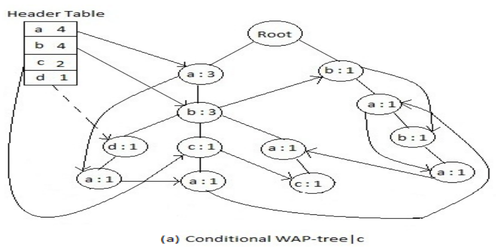


Fig. 2 Reconstruction of WAP tress for mining conditional pattern base c

Using these conditional sequences, a conditional WAP tree, WAP-tree|c, is built using the same method which was used previously. The new conditional WAP-tree is shown in figure 4.2(a). Recursively, based on the WAP-tree in figure 4.2(a), the next conditional sequence base for the next suffix subsequence, cc is found as ab(1), aba(1).

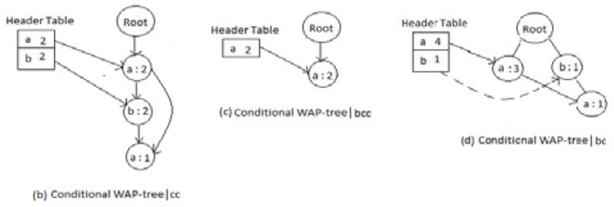


Fig. 3 Reconstruction of WAP tress for mining conditional pattern base c

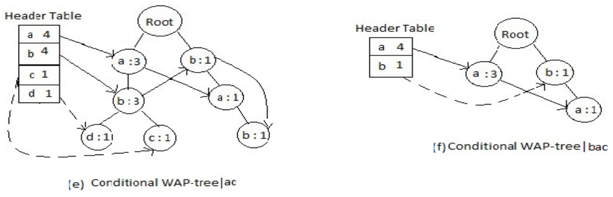


Fig. 4 Reconstruction of WAP trees for mining conditional pattern base c.

With these sequence obtained is used to construct the next WAP tree shown in figure 4.2(b). Next conditional sequence base for suffix bcc as a:2. This ends the re-construction of WAP trees that progressed as suffix sequences |c, |cc, |bcc, |acc and the frequent patterns found along this line are cc, bcc, abcc, acc. The recursion continues with the suffix path |c, |bc. Thus, the conditional WAP-tree for |bc is built as shown figure 4.2(d) The algorithm keeps running, finding the conditional sequence bases of ac as abd:1, abc:1, ab:1 and bab:1. Then, the conditional WAP-tree|ac is built as shown in figure 4.2(e). This has header count of 4, 4, 1, 1 for event a, b, c, d respectively. Since the count of c & d is less than the minimum support threshold, it is discarded in future mining process. The conditional search of c is now finished. The search for frequent patterns that have the suffix of other header frequent events starting with suffix base |b and then |a but not d as it does not meet support threshold are also mined the same way the mining for patterns with suffix c is done above. Discovered frequent pattern set for c is: {c, cc, acc, bcc, abcc, aac, bac, abac, ac, abc, bc}.The conditional search of c is now finished. The search for frequent patterns that have the suffix of other header frequent events (starting with suffix base |b and then |a) are also mined the same way the mining for patterns with suffix c is done above. After mining the whole tree, discovered frequent pattern set is: { c, cc, acc, bcc, abcc, aac, bac, abac, ac, abc, bc, b, ab, a, aa, ba, aba}

3. Performance Analysis

Experimental datasets are generated by using our data generation program. The datasets consist of sequences of events, where each event represents an accessed web page. The parameters shown below are used to generate the data sets.

|D|: Number of sequences in the database

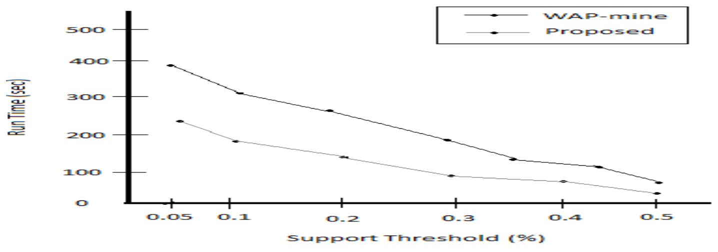
|C|: Average length of the sequences

|N|: number of events

For example, C10· N50·D10 K means that |C| = 10, |N| = 50, and |D| = 10K. It represents a group of data with average length of the sequences as 10, the numbers of individual events in the database are 50, and the total number of sequences in database is 10 thousand. The datasets with different parameters test different aspects of the algorithms. Basically, if the number of these three parameters becomes larger, the execution time becomes longer.

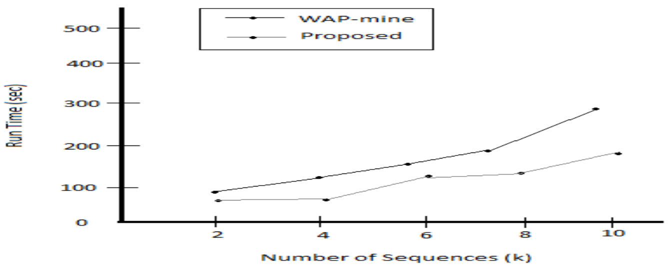
Experiment 1: Execution times for different support threshold:

Fixed size database and different minimum support is used in this experiment to compare the performance of WAP and proposed algorithms. The datasets are described as C15 · N50 · D10 K, and algorithms are tested with minimum supports between 5% and 50% against the 5 thousand (5 K) database.



Experiment 2: Execution times for databases with different size:

Databases with different sizes from 2 K to 10 K with the fixed minimum support of 30% were used in this experiment. The five datasets are C10 · N40 · D2 K, C10 · N40 · D4 K, C10 · N40 · D6 K, C10 · N40 · D8 K, and C10 · N40 · D10 K. Performance measurement of the WAP and Proposed algorithms were compared and the results of this experiment are presented in figure:



4. References

[1] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu. 2000. Mining access patterns e_ciently from web logs. In Proceedings of the Paci_c-Asia Conference on Knowledge Discovery and Data Mining (PAKDD00). Kyoto,Japan, pp. 396-399, 400-402, 2000

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In Proceedings of the 20th International Conference on Very Large Databases. Santiago, Chile, pp. 487499,1994.

[3] R. Kosala, H. Blockeel: Web Mining Research: A Survey. ACM SIGKDD Explorations,Vol. 2. (2000) 1-15

[4] C. EZEIFE and Y. LU. Mining Web Log Sequential Patterns with Position Coded Pre-Order LinkedWAP-Tree. International Journal of Data Mining and Knowledge Discovery(DMKD) Kluwer Publishers, pp. 6, 12-15, 27-34, 2005.

[5] B. Zhou, S. Cheung Hui, and A. Fong. CS-Mine: An E_cient WAP-Tree Mining for Web Access Patterns. School of Computer Engineering, Nanyang Technological University, Singapore, pp. 523- 527, 530-532.

[6] R. Cooley, B. Mobasher, and J. Srivastava: Data Preparation for Mining World Wide Web Browsing Patterns. Journal of Knowledge and Information Systems, Vol. 1, No. 1, pp. 67-71, 1999.

[7] E. Maged, A. R. Elke and R. Carolina: FS-Miner: An Efficinet and Incremental System to Mine Contiguous Frequent Sequences. Computer Science Technical Report Series,Worcester Polytechnic Institute pp. 128-130, 2003.

[8] R. Cooley, P.-N. Tan, J. Srivastava. Discovery of Interesting Usage Patterns from Web Data. Proceedings of the International Workshop on Web Usage Analysis and User Pro_ling, pp. 163182. 1999.

[9] J. Parmar and S. Garg. Modi_ed Web Access Pattern (mWAP) Approach for sequential Pattern Mining. International Conference on Data Mining, pp. 30-35.