Computer and Information Science
www.ccsenet.org/journal.html
Vol. 1, No. 4
November 2008

Web Access Pattern Algorithms in Education Domain

C. Gomathi (Corresponding Author)
P.G. Department of Computer Science
Kongu Arts and Science College
Erode-638-107, Tamil Nadu, India
E-mail: kc.gomathi@gmail.com

M. Moorthi
Department of Computer Science
Kongu Arts and Science College
Erode-638-107, Tamil Nadu, India
E-mail: moorthi_bm_ka@yahoo.com

Dr. K. Duraiswamy
K.S. R. College of Technology
Tiruchengode- 637-209, Tamil Nadu, India
E-mail: kduraiswamy@yahoo.co.in

Abstract

Sequential pattern mining discovers frequent user access patterns from web logs. Apriori-like sequential pattern mining techniques requires expensive multiple scans of database. So, now days, WAP (Web Access Pattern) tree based algorithm is used. It is faster than traditional techniques. However, the use of conditional search strategies in WAP-tree based mining algorithms requires re-construction of large numbers of intermediate conditional WAP-trees, which is also very costly.

In this paper, Kongu Arts and Science College (KASC) web logs are taken for mining. Here, we propose an efficient sequential pattern mining techniques for KASC web log access sequences known as CS-WAP Tree. This proposed algorithm modifies the WAP tree approach for improving efficiency. The proposed algorithm totally eliminates the need to engage in numerous reconstructions of intermediate WAP trees and considerable reduces execution time. The results of experiments show the efficiency of the improved algorithm. The next key aim is to compare WAP algorithms.

Keywords: Web log file, Data mining, Web Usage Mining, Web Access Pattern

1. Introduction

Web usage mining [2] is used to identify user behavior on a particular website. It performs mining on web usage data or web logs. The mined knowledge can then be used in many practical applications [4], such as improving the design of web sites, analyzing user behaviors for personalized services, and developing adaptive web sites according to different usage scenarios.

The access and usage of information is the major functionality of the broad diversity of WWW user community. As a result, discovery of useful information from Web content is not the only important task of Web mining. Resource finding, information selection and pre-processing, generalization and analysis are sub-tasks of Web mining. Web mining is a broad domain which covers fields like database, information retrieval and artificial intelligence, machine learning, natural language processing, network analysis and information integration.

A sequential access pattern [3] is a sequential pattern in a large set of pieces of web logs, which is pursued frequently by users. Most of the previous studies for discovering sequential patterns are mainly based on the Apriori algorithm [1]. However, these algorithms encounter the same problem as most Apriori-based algorithms.

The WAP-mine algorithm [5] has been developed for mining sequential access patterns from the WAP-tree. This approach avoids the problem of generating an explosive number of candidates as encountered in Apriori-based algorithms. The WAP-mine algorithm is in general an order of magnitude faster than traditional sequential pattern mining techniques. In this paper, we propose an efficient sequential access pattern mining algorithm known as Constrained Sequence WAP mining algorithm (CS-WAP).

2. Background Study

As an important branch of data mining, sequential pattern mining, which finds high-frequency patterns with respect to time or other patterns, was first introduced by (R. Agrawal, & R. Srikant (1994)) as follows: given a sequence database where each sequence is a list of transactions ordered by transaction time and each transaction consists of a set of items, find all sequential patterns with a user specified minimum support, where the support is the number of data sequences that contain the pattern. Since the access patterns from web log take on obvious time sequence characteristic, it is natural to apply the technology of sequential pattern mining to web mining. According to the downward closure property of frequent sequences, to some extent, maximal frequent sequences have already included all frequent sequences. The space to store maximum frequent sequences is much lower than to store complete set, and web mining applications partly only depend on maximum frequent sequences rather than the complete set of frequent sequences so that mining maximum frequent access sequences is of essential practicability.

Sequential access pattern mining techniques are mainly based on two approaches: Apriori-based mining algorithms and WAP tree based mining algorithms.

2.1 Apriori-based Mining Algorithms

The AprioriAll (J. Pei, J. Han, B. Mortazavi-asl & H. Zhu (2000)) algorithm proposed a three-step approach for mining sequential patterns. It first finds all frequent itemsets. Then, it transforms the database such that each original transaction is replaced by the set of all frequent itemsets contained in the transaction. And finally, it finds the sequential patterns. However, this algorithm does not scale well due to the costly transformation step. In (J. Pei, J. Han, B. Mortazavi-asl & H. Zhu (2000)), a generalized sequential pattern mining algorithm known as GSP mining algorithm was proposed. Similar to the AprioriAll algorithm, GSP scans the database several times. In the first scan, it finds all frequent items and forms a set of frequent sequences of length one. In subsequent scans, it generates candidate sequences from a set of frequent sequences obtained from the pervious scan and checks their supports. The process terminates when no candidate is found to be frequent. So this algorithm requires multiple scans of database. So now we discuss WAP tree based mining algorithm.

2.2 WAP Mining Algorithm

The WAP-tree is a very effective compressed data structure designed for storing the data obtained from web logs. To construct a WAP-tree, we need two scans of the web access sequence database: (1) Scan database once, find all frequent individual events; (2) Scan database again, construct the WAP-tree over the sub-sequences with only frequent individual events of each sequence, which are also called frequent subsequences, by merging their common prefixes,. At the same time, all nodes that contain the same frequent event are linked into an event queue and the Header Table with all frequent events is created for this WAP-tree with the head of each event queue registered in it. Then, all the nodes labeled with the same event can be visited by following the related event queue, starting from the Header Table.

The FS-tree extends the WAP-tree structure for incremental and interactive mining. The corresponding mining algorithm FS-mine (Frequent Sequence mining) is used to analyze the FS-tree to discover frequent sequences.

J Han puts forward a web access pattern tree structure (WAP-tree) and an algorithm for mining frequent access path based on WAP-tree (WAP-mine) in (J. Pei, J. Han, B. Mortazavi-asl & H. Zhu (2000)). This algorithm and not producing candidate frequent patterns. Consequently, WAP-mine algorithm is an order of magnitude faster than Apriori algorithm (B.Y. Zhou, S.C. Hui, & A.C.M. Fong (2004)) put forward by Agrawal at earlier stage. Nevertheless, WAP-mine needs to produce a mass of conditional WAP-tree, which influences the efficiency of WAP-mine in a certain degree. In recent years, some classical algorithms applied to mine maximum patterns include MaxMiner, DethProject, MAFIA and GenMax etc.

3. Prototype Description - CS – WAP

The CS – WAP mine (Constrained Sequence Web Access Pattern mining algorithm) enhances our WAP-tree based algorithm (J.Srivastava, R.Cooley, M. Deshpande, & P.-N. Tan (2000)) by employing directly the *conditional sequence base* of each frequent event, **without the need of constructing any WAP trees**. Although, the WAP-tree is a highly compressed data structure for storing sequence data, we need recover the uncompressed sequences during mining process in practice. That is the reason why our new CS-WAP mine algorithm is not based on the WAP-tree as our previous CS-mine algorithm. The proposed CS-WAP mine algorithm can improve quite significantly on the efficiency

when compared with the WAP-mine algorithm, especially when the support threshold becomes smaller and the size of database gets larger.

The Prototype Algorithm

The prototype algorithm for mining sequential access patterns using KASC web log is shown in Figure 1.

Input: Web access sequence database, Minimum Support Threshold

Output: Set of sequential access patterns

Method:

Step 1 : Construct event queues for *CSB*(*Sc*).

Step 2 : Test single sequence for *CSB*(*Sc*).

(i) If test is successful, insert all ordered combinations of items in frequent sequence *FS* = *Sc*+*SingleSeq* into *SAP*.

(ii) Otherwise, for each *ej* in Header Table of *CSB*(*Sc*), use **SubQueue** to construct *CSB*(*Sc*+*ej*). Set *Sc* = *Sc*+*ej* and recursively mine *CSB*(*Sc*) from step 3.

Step 3 : Return *SAP*.

Figure 1. The CS-WAP algorithm

4. Experimental Results

The proposed algorithm is implemented in VB.NET and all experiments were found on Intel Pentium running on Microsoft Window XP profession. The web server log file dated on Sep 2007 from KASC web server after preprocessed (Gomathi.C., Moorthi M. & Duraiswamy K. (2008)) has been selected for our experiments. This KASC log file size is 148KB. The proposed method was applied on this preprocessed web log files to prepare sequence pattern. The proposed mining algorithm has significant advantages when compared with the original WAP-mine algorithm.

From table-1 and Figure 2, It can be seen that the execution time increase as size of the database increase. As the size of the data decrease, the execution times also decrease. So the CS - WAP algorithm always use less run time than other WAP algorithms. Also the table shows that WAP tree algorithm requires more memory than proposed algorithm. It reduces storage cost. The experiments showed that the proposed methodology needs less time to find frequent sequence and needs only minimum storage area.

5. Conclusion

In this paper, CS-WAP mine tool developed using VB.NET for sequential access pattern from KASC web log files. The proposed system eliminates the need to store numerous intermediate WAP trees during mining. Since only the original tree is stored, it drastically cuts off huge memory access costs. This system also eliminates the need to store and scan intermediate conditional pattern bases for reconstructing intermediate WAP trees. This algorithm uses the pre-order linking of header nodes to store all events *ei* in the same suffix tree closely together in the linkage, making the search process more efficient.

"The proposed algorithm analyzes the student's behaviors. Based on the behaviors, the web site is restructured to get knowledge immediately. So, student's society updates their knowledge, face challenge and find the solutions for it".

References

R. Agrawal, and R. Srikant. (1994). Fast Algorithms for Mining Association. *In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, Santiago, Chile, pp. 487-499.

R. Kosala, and H. Blockeel. (2000). Web Mining Research: A Survey. *In ACM SIGKDD Explorations*, Vol.2, pp. 1-15.

Gomathi.C., Moorthi M., Duraiswamy K. (2008). Preprocessing of Web Log Files in Web Usage Mining. *The ICFAI journal of Information Technology,* Vol. 4, No. 1, pp. 55-66.

J. Pei, J. Han, B. Mortazavi-asl, and H. Zhu. (2000). Mining Access Patterns Efficiently from Web Logs. *In Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Kyoto, Japan, pp. 396- 407.

J.Srivastava, R.Cooley, M. Deshpande, and P.-N. Tan. (2000). Web Usage Mining:Discover and Applications of Usage Patterns from Web Data. *In ACM SIGKDD Explorations*, Vol. 1, No. 2, pp.12-23.

B.Y. Zhou, S.C. Hui, and A.C.M. Fong (2004) CS-mine: An Efficient WAP-tree Mining for Web Access Patterns. *In Proceedings of the 6th Asia Pacific Web Conference (APWeb'04),* Hangzhou, China, Lecture Notes in Computer Science 3007, Springer, pp. 523-532.

Table 1. Execution times trend with different data sizes with fixed minimum support 7%

	Different Changed transaction size (navigational path)		
Algorithms time in second	63K	126K	189K
WAP	360	600	780
Efficient-WAP	300	420	480
CS-WAP	60	180	240

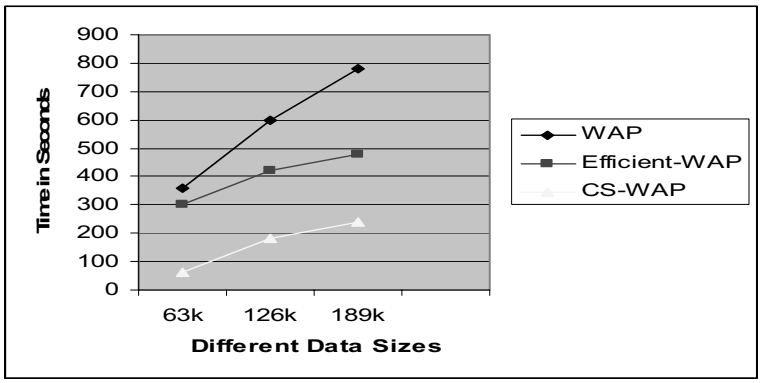

Figure 2. Execution time s trends with different data sizes.