SEQUENTIAL PATTERN MINING FROM WEB LOG DATA

Rajashree Shettar

1Associate Professor, Department of Computer Science, R. V College of Engineering, Karnataka, India, rajashreeshettar@rvce.edu.in

Abstract
Sequential Pattern Mining involves applying data mining methods to large web data repositories to extract usage patterns. The growing popularity of the World Wide Web, many websites typically experience thousands of visitors every day. Analysis of who browsed what, can give important insight into the buying pattern of existing customers. Correct and timely decisions made based on this knowledge have helped organizations in reaching new heights in the market. In this paper, the sequence tree algorithm is implemented for pattern mining and this is experimented on web log data. The web log data which is considered as secondary data of the web has been considered for the discovery of frequent sequential patterns. The results have shown that the sequence tree algorithm performs better than the well-known Generalized Sequential Pattern (GSP) algorithm. The experiment shows that the running time of sequence tree algorithm is faster than the standard GSP algorithm and also Sequence Tree algorithm discovers more number of patterns than the standard GSP algorithm.

Index Terms: Web usage mining, sequential patterns, sequence tree, web log data.

1. INTRODUCTION
Knowledge extraction from the World Wide Web has become an important and challenging task as enormous amount of data in form of semi-structured nature is available. Web mining approaches such as web content mining, web structure mining and web usage mining are available [1, 2, 3] which help to extract and produce useful knowledge from the web. Web usage mining is used for applications such as customer shopping sequences, web clicks, biological sequences, creation of dynamic user profiles, etc. In this paper, the discovery of frequent patterns from web log data is being considered. A web server usually registers a log entry for every access of a web page. First, raw web log data needs to be cleaned, condensed and transformed in order to retrieve and analyze significant and useful information. Second, pattern mining can be performed on log records to find association patterns, sequential patterns and trends of web accessing. With the use of such patterns, studies have been conducted on analyzing system performance, improving system design by web caching, web page pre-fetching, understanding the nature of web traffic, etc. The process of web usage mining consists of three major steps [4] (1) data pre-processing, (2) pattern discovery, and (3) pattern analysis stages. Log files are stored on the server side, on the client side and on the proxy servers. In this work, only the server side web log data is being used which is the Click stream data set [5]. The data pre-processing phase includes the data cleaning, user identification, session identification and data transformation respectively. The pattern discovery phase involves the discovery of frequent sequences. The pattern analysis phase involves the analysis of the frequent patterns generated by the pattern discovery phase.

Many different techniques for mining frequent sequential patterns from the log data have been proposed in the recent past. The authors in [10] discuss that the candidate-generation-and-test approach outperforms the pattern-growth approach on mining short patterns, while pattern-growth approach is better on mining long patterns. The authors in [11] discuss the process of web log mining. Sequence mining is accomplished in [12], where a so-called WAP-tree is used for storing the patterns efficiently. Tree-like topology patterns and frequent path traversals are searched by [13, 14, 15, 16].

2. DATA PRE-PROCESSING
Data pre-processing phase involves data cleaning, user identification and session identification and data transformation [4]. In data cleaning stage phase, the web log is examined and irrelevant or redundant items such as image, sound, video files, executable cgi files and HTML files which could be downloaded without an explicit user request are removed. Data cleaning stage also involves removal of
HTTP errors, records created by crawlers, etc. Data cleaning is performed by checking for file extensions such as GIF, JPEG, jpg, mpg, avi, etc. These are removed from the log. Each record of the log contains the following 6 fields. They are the shop-id, time the page was visited, IP address of the user visiting the page, a unique session identifier, the current page that is visited and the last field is the referrer which refers to the page from which the current page was visited.

The user identification phase involves identification of users from the log data. The following procedure is adopted for identifying the users. (1) A new IP identifies a new user. (2) If the same IP is used, but different web browsers or different operating system in terms of type and version is being used, then this is considered as a new user.

The session identification stage involves identifying sessions according to different users. A session is a group of activities performed by the user while navigating through a given site within a given time period. In this paper, a set of pages visited by a specific user is considered as a single user session if the pages are requested at a time interval not larger than a specified time period of 60 milliseconds.

In the data transformation phase, the web log data is converted into a format needed by the mining algorithms. For every session that is identified, the log data is converted into two databases – transactional database and sequential database. The transactional database consists of two fields – transaction id and the set of pages visited in a session. A sequential database consists of sequence of pages visited in a session and the data set used is Click Stream data which is server side log data. The records are sorted in ascending order based on IP address and the number of users is identified. The boundaries of each user are marked and stored in a two dimensional array. The boundaries specify the starting index and the ending index of a particular user in a sorted log file. In the session identification phase of the pre-processing stage, a session timestamp value is set which specifies the total time of a particular session. The value of the session timestamp has been set as 60 milliseconds. The record obtained is checked to see if it belongs to a particular session. If it belongs to the same session, it is added to the session list and the next record is processed. If it does not belong to the same session, then the previous session records are put onto the file and then the session list is cleared. A new session is created and the current record is added to the new session and its timestamp value is initialized as the session start time. This process is repeated for every user and all sessions are identified. The records in every session are then transformed into item sets and sequences. In the data transformation phase, all the fields of the records are transformed into appropriate format.

3. FREQUENT SEQUENCES

Sequential Pattern mining involves discovering frequent sequences from a database where data to be mined is in some sequential order, this was first introduced by Agrawal and Srikant [6], and from then on the goal of sequential pattern mining is to discover all frequent sequences of itemsets in a dataset. In particular, an itemset is a non-empty subset of elements from a set C, the item collection, called items. In this manner, an itemset represents the set of items that occur together. The itemset composed of items a and b is denoted by (ab). A sequence is an ordered list of itemsets. A sequence is maximal if it is not contained in any other sequence. A sequence with k items is called a k-sequence. The number of elements (itemsets) in a sequence s is the length of the sequence and is denoted by |s|. The i-th itemset in the sequence is represented by s_i, and the set of considered sequences is usually designated by database (DB), and the number of sequences by database size (|DB|). A subsequence s’ of s is denoted by s’ ⊆ s. Formally, a sequence a = <a_1, a_2, a_3, ..., a_n> is a subsequence of b = <b_1, b_2, b_3, ..., b_m>, if there exists integers 1 ≤ i_1, i_2, i_3, ..., ≤ m such that a_1 ≤ b_1, a_2 ≤ b_2, a_3 ≤ b_3, ..., a_n ≤ b_m [8].

Let I = {a_1, a_2, ..., a_m} be a set of items, and a transaction database DB = {T_1, T_2, ..., T_n}, where T_i (i ∈ [1, n]) is a transaction which contains a set of items in I. The support (or occurrence frequency) of a pattern A, is the number of transactions containing A in DB, where A is a set of items. A pattern A is frequent if A’s support is no less than a predefined minimum support threshold, θ. Given a transaction database DB and a minimum support threshold θ, the problem of finding the complete set of frequent patterns is called the frequent-pattern mining problem. [9]

4. SEQUENCE TREE ALGORITHM

Sequential tree algorithm is used to find the frequent sequences in the log file. It involves two stages- construction of the sequence tree and mining of the sequence tree for finding frequent sequences. It takes as input the sequential database records which specify the various sequences of pages visited by the user in a particular session and also threshold. The frequent sequences that have been identified from the log file based on the minimum support threshold are obtained as output. The sequence tree algorithm is described in section 3.1 from steps (1) to (6).
4.1 Algorithm Sequence Tree

(1) Read the sequential database and create a map with key-value pairs. Key refers to the unique page name that was visited and sequence value refers to the frequency or number of times the page occurs.
(2) Sort the map in descending order of frequency of the keys.
(3) Create a sequence tree using the following steps.
   A root node is termed as null.
   For every row of sequential database that is read.
   Attach the sequence as a branch to the root.
   If the sequence is already present increment the counter.
   else a new branch is created with that sequence.
(4) Include a header table with number of rows equal to sequences with value one. Each row of header table is a linked list of nodes which specify the position where nodes are present.
(5) Perform mining process.
   For every row in the header table.
      If frequency < minimum support threshold.
         Ignore the row of header table.
      else.
         For every node in the particular linked list.
            If frequency < minimum support threshold.
               Ignore the path.
            else.
               Traverse the tree up till the root and store the path in the file as frequent sequences.
      EndFor.
   EndFor.
(6) The frequent sequences identified from the algorithm are stored in a file.

5. EXPERIMENTAL RESULTS

The experimental results and analysis of frequent sequences discovered from web log data is described in this section. The experimental analysis is conducted on the Click Stream Data set [5] which is a server side web log data containing 12,000 records. Each record contains the following fields – a shop identifier, time stamp, IP address, unique session identifier, page visited, and referer. The performance is tested on a computer with a 1.41GHz processor. The program is developed using JDK 1.6. Table 1 shows the comparison between the running times of GSP[7] and Sequence tree algorithms for different threshold values. Table 2 shows the comparisons between number of patterns generated by GSP and Sequence tree algorithms for different threshold values.

Graph 1 shows the comparison of running time of GSP and Sequence tree algorithm for different threshold values. Graph 2 shows the number of patterns found by GSP and Sequence tree algorithms for different threshold values.

**Table 1: Comparison between the running times of GSP and Sequence tree algorithms**

<table>
<thead>
<tr>
<th>Threshold Values</th>
<th>GSP Algorithm (in mS)</th>
<th>Sequence Tree Algorithm (in mS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>44444</td>
<td>47</td>
</tr>
<tr>
<td>20</td>
<td>35146</td>
<td>47</td>
</tr>
<tr>
<td>30</td>
<td>38860</td>
<td>47</td>
</tr>
<tr>
<td>50</td>
<td>37690</td>
<td>32</td>
</tr>
<tr>
<td>80</td>
<td>37501</td>
<td>31</td>
</tr>
</tbody>
</table>

**Table 2: Comparisons between number of patterns generated by GSP and Sequence tree algorithms**

<table>
<thead>
<tr>
<th>Threshold Values</th>
<th>GSP Algorithm</th>
<th>Sequence Tree Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>179</td>
<td>1568</td>
</tr>
<tr>
<td>20</td>
<td>70</td>
<td>658</td>
</tr>
<tr>
<td>30</td>
<td>34</td>
<td>374</td>
</tr>
<tr>
<td>50</td>
<td>11</td>
<td>183</td>
</tr>
<tr>
<td>80</td>
<td>5</td>
<td>107</td>
</tr>
</tbody>
</table>
Graph 2: The number of patterns found by GSP and Sequence tree algorithms

It is observed that the running time taken by Sequence tree algorithm is lesser than that of GSP algorithm. The total number of patterns generated by Sequence tree algorithm is greater than the GSP algorithm.

6. CONCLUSION

Web usage mining techniques has been applied to large web repositories to extract usage patterns. The sequence tree algorithm is one such web usage mining technique which extracts frequent sequential patterns by formation of a tree. The running time and number of patterns generated are observed. The experimental results show that the Sequence Tree algorithm shows faster running time than the standard GSP algorithm and it also discovers more number of patterns than the standard GSP algorithm.

REFERENCES


http://lisp.vse.cz/challenge/CURRENT


[8] Cláudia Antunes and Arlindo L. Oliveira Sequential “Pattern Mining Algorithms: Trade-offs between Speed and Memory”.

[9] Jiawei Han, Jian pei, Yiwen yin and Runying mao, “Mining Frequent Patterns Without Candidate Generation: A Frequent Pattern Tree Approach”, Data Mining and Knowledge Discovery, 8, 53–87, 2004.


