Efficient Incremental Mining of Qualified Web Traversal Patterns without Scanning Original Databases

Jia-Ching Ying¹, Vincent S. Tseng¹
¹Dept. of Computer Science and Information Engineering
National Cheng Kung University
Tainan, Taiwan, R.O.C.
tsengsm@mail.ncku.edu.tw

Philip S. Yu²
²Department of Computer Science
University of Illinois at Chicago
Illinois, U.S.A.
psyu@cs.uic.edu

Abstract—Discovering web traversal patterns is an important issue in web usage mining with various applications like navigation prediction and improvement of website management. Since web data grows so rapidly and some web data may become out of date over time, we need not only consider the new data but also delete the old one to re-mine new web traversal patterns. To reduce the overhead of re-mining the web traversal patterns from the whole web data, an incremental mining approach is needed by using the previous mining results and computing new patterns just from the inserted or deleted part of the web data. In this paper, we propose an efficient incremental web traversal pattern mining algorithm named IncWTP_PLM (Incremental mining of Web Traversal Patterns by using Projected-database Link Matrix). Meanwhile, a special data structure named Projected-database Link Matrix is proposed to avoid scanning original database. Besides, the website structure is also considered in IncWTP_PLM such that each web traversal pattern discovered is qualified. The experimental results show that our algorithm outperforms other approaches substantially in terms of efficiency.

Keywords—Web mining; incremental mining; qualified web traversal pattern; navigation prediction

I. INTRODUCTION

In recent years, vast amount of data would be easily produced and collected from the web environment because of the growth of internet. Hence, how to discover the useful information and knowledge efficiently from these vast amounts of web data has become a more and more important topic recently.

Web mining [1][3][4][5][8][10] is present to improve the web services by extracting useful information and knowledge from vast amount of web data. Among the previous studies [4][5], web traversal patterns can be used to represent the browsing behaviors of most users in a website. Thus, mining web traversal patterns could improve the website design by discovering most of users’ access patterns from web data. For example, we could use the web traversal patterns to provide efficient access between two pages with high dependency, and better design for web pages, and so on. Furthermore, these patterns also could improve marketing decisions, like advertisements recommendation, customer classification, behavior analysis, and so on. However, the web data grows rapidly and some of the user sequences may be out of date over time. Therefore, the web traversal patterns should be updated when the new web data are inserted into and the old web data are deleted from original web traversal sequence database.

A number of studies have been done on incrementally mining web traversal patterns from large amount of web data. El-Sayed et al. [3] presented FS-Miner algorithm that preserves some necessary information in a tree structure, named frequent sequence tree structure (FS-tree). Unfortunately, the purpose of El-Sayed et al.’s work just only extract frequent sequence patterns that can be used for pre-fetching and caching. For pre-fetching and caching, knowledge of such ordered contiguous page references is useful for predicting future references [3]. Thus, the FS-tree could be compact for fitting into memory. However, some important web traversal patterns which are not ordered contiguous page references might be lost. For example, if <AB> is such web traversal pattern which would be loss in FS-Miner, that is there are many sequences in web data containing the pattern <AB> but A and B are not often consecutive pages in the sequences. However, the pattern <AB> has been meaningful for website owner to refine the website and analyzing users’ behavior.

IncWTP algorithm [4] proposed by Lee et al. used a lattice structure to keep the candidate sequences whose support counts are greater than zero into memory to improve efficiency of incremental mining. Moreover, in [4], Lee et al. proposed new viewpoint of mining web traversal patterns, considering website structure. Such web traversal patterns are called “qualified” web traversal patterns. Actually, when the website owner refines the website or analyzed users’ behavior the website structure should be considered at the same time. To avoid the web traversal patterns which contravene the website structure, IncWTP algorithm check the website structure while generating candidate sequences. However, it is very time-consuming for incremental step to generate new candidate sequences and scan whole web data (new and original web dataset). If there are new qualified web traversal pattern with length k discovered from the updated database, it may generate several new candidate sequences with length k+1. Hence the algorithm must scan whole web data (new and original web dataset) more than once.

Although these studies have provided useful algorithms and valuable viewpoints, there exist still many shortcomings. FS-Miner algorithm is efficient but would loose important information for website owners. IncWTP algorithm utilizes a lattice structure and the web site structure to reduce the mining time and space. However, it is still not efficient
enough since it is an apriori-like algorithm. Besides, it probably needs to scan the original database because of new candidates generated when the database is updated.

In this paper, we propose an efficient incremental qualified web traversal pattern mining algorithm named IncWTP_PLM (Incremental mining of qualified Web Traversal Patterns by using Projected-database link matrix), for re-mining all the qualified web traversal patterns when the database is updated. In this paper, Projected-database link matrix and projected database are selected as our storage structure for IncWTP_PLM algorithm to store previous mining results. The projected-database link matrix is kept in memory and the projected database is stored in hard disk. That is, all of the necessary information in original web dataset is stored in these two kinds of storage. Hence, IncWTP_PLM algorithm can avoid unnecessary scanning of the original web dataset. Thus, I/O cost could be frugal. Through a series of experiments, IncWTP_PLM algorithm was shown to outperform IncWTP algorithm in most cases, especially when minimum support is set small. For instance, IncWTP_PLM algorithm is about 3–5 times faster than IncWTP algorithm under experiments where the minimum support is set as 0.8%.

The rest of this paper is organized as follows. In Section 2, we give the definition of the targeted problem. The related work is given in Section 3, and our method, IncWTP_PLM, is described in details in Section 4. In Section 5, the experimental evaluation results are explained, and a brief conclusion with future work is given in Section 6.

II. PROBLEM DEFINITION

Formally, we define several terminologies as follow:

Definition 1. A website structure is a weak connected directed graph \( W \) consisting of a page set \( P(W) = \{p_1, p_2, \ldots, p_m\} \) as the vertex set of the directed graph \( W \) and out-link set \( L(W) = \{p_i, p_j | p_i, p_j \in P(W), 1 \leq i \leq n, 1 \leq j \leq n, i \neq j\} \) as the directed edge set of the directed graph \( W \), where \( p_i, p_j \) is belong to \( L(W) \) means \( p_i \) has an out-link to \( p_j \).

Definition 2. A web traversal sequence \( S = \{p_1, p_2, \ldots, p_n\} \), \( \forall p_i \in P(W), \text{where} \ 1 \leq i \leq m \) is a list of web pages which is ordered by traversal time, and each web page can repeatedly appear in a web traversal sequence.

Definition 3. A qualified web traversal sequence \( S = \{p'_1, p'_2, \ldots, p'_{m}\} \) is a web traversal sequence, \( \forall p'_j, p'_{j+1} \in L(W), \text{where} \ 1 \leq j \leq m-1 \).

Definition 4. A web traversal sequence database \( D = \{\{\text{Sid}_1, S_1\}, \ldots, \{\text{Sid}_k, S_k\}\} \) is a database consisting of web traversal sequences, where \( \text{Sid}_i \) is the session ID of web traversal sequence \( S_i \), \( 1 \leq i \leq k \).

Definition 5. Given a threshold \( \tau \), named minimum support, and a web traversal sequence database \( D \), a qualified web traversal pattern \( P \) is a qualified web traversal sequence and the percentage of web traversal sequences in \( D \) which are supersequence of \( P \) is greater \( \tau \).

For example, if we set minimum support as 40%, TABLE I. as the web traversal sequence database and Figure 1. as the website structure, the pattern \( <b,e> \) should be found, because it is contained in the sequences \( (1, <bgiabehi>), (2, <bgabe>), (5, <dcabcbehi>), (6, <cebdhdia>), (7, <abefdcda>), (8, <abefdcda>), (9, <bie>) \) and there is a link from page b to page e in the website structure. But the pattern \( <b,i> \) should not be found, because there is no link from page b to page i in the website structure.

### TABLE I. An example of web traversal sequence database.

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sequence</th>
<th>Sid</th>
<th>Sequence</th>
<th>Sid</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>bgiabehi</td>
<td>5</td>
<td>dcabcbehi</td>
<td>8</td>
<td>abefdcda</td>
</tr>
<tr>
<td>2</td>
<td>bgabe</td>
<td>6</td>
<td>cbehdia</td>
<td>9</td>
<td>bie</td>
</tr>
<tr>
<td>3</td>
<td>acabg</td>
<td>7</td>
<td>abefdcda</td>
<td>10</td>
<td>igbcd</td>
</tr>
<tr>
<td>4</td>
<td>acdcabgi</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 1. An example of website structure](image)

Definition 6. An inserted web traversal sequence database is a new web traversal sequence database that we insert into original web traversal sequence database.

Definition 7. A deleted web traversal sequence database is a subset of old web traversal sequence database that we delete from original web traversal sequence database.

Definition 8. An incremental web traversal sequence database is a web traversal sequence database consisting of inserted web traversal sequence database and deleted web traversal sequence database.

We assume that a web data is already transformed to web traversal sequence database before mining as follow TABLE I., and the website structure is also given as above Figure 1. Our goal is to discover the qualified web traversal patterns incrementally from a web traversal sequence database. Since the qualified web traversal patterns we found are frequent, we should given a threshold, minimum support, to stem infrequent patterns.

III. RELATED WORK

In fact, qualified web traversal pattern mining has a great resemblance to sequential pattern mining [2][6][7][9][11]. The most significant difference between qualified web traversal patterns and sequential patterns is that each two consecutive web pages in a qualified web traversal pattern there must be a link from first to the second in the web structure. On the contrary, sequential pattern mining only discovers sequential patterns from a sequence database. Whether incremental sequential pattern mining or incremental web traversal pattern mining, the storage structure is the key of the efficiency improvement. We then introduce the previous studies for incremental sequential
pattern mining or incremental web traversal pattern mining classed with their storage structures. First, we introduce the two lattice based incremental sequential pattern mining and incremental qualified web traversal pattern mining algorithm. An incremental sequential pattern mining algorithm ISL (Incremental Sequence Lattice Algorithm) was proposed by Zaki et al. [9], in which, the algorithm based on SPADE algorithm [11] using equivalence classes to discovery sequential patterns. The lattice structure keeps all sequential patterns, candidate sequences, and their support counts in memory. When the database is updated, ISL algorithm updates the lattice structure and then re-mines new sequential patterns form the updated lattice. Because only newly generated candidate sequences need to be counted from the original database, the mining efficiency is improved.

However, ISL algorithm also keeps the candidate sequences whose support counts are zero in the lattice, it would cause the lattice structure is too huge to fit into memory. Besides, ISL only addresses the insertion of transactions into the pre-existing user sequences. This is because it makes efforts of processing the new user sequences are inserted into the customer sequence database. To improve these shortcomings, IncWTP algorithm was proposed by Lee et al. IncWTP algorithm uses a special lattice structure, named extended lattice structure \(^1\), to keep the candidate sequences whose support counts are greater than zero. In addition, the sequence id also is stored in the lattice structure to improve the efficiency during performing the incremental mining process. When the new web traversal sequences are inserted into the web traversal sequence database, or old web traversal sequences are deleted from the original web traversal sequence database, IncWTP maintains the structure for mining the new qualified web traversal patterns.

Another famous storage structure of incremental sequential pattern mining is projected-database. An incremental sequential pattern mining algorithm based on PrefixSpan algorithm [6][7] was proposed by Cheng et al. [2], named IncSpan algorithm (incremental mining in sequential pattern algorithm). IncSpan algorithm uses the concept of a projected-database to recursively mine sequential patterns.

Unfortunately, ISL and IncSpan algorithms only address the insertion of transactions into the pre-existing user sequences. For above reasons, IncWTP is more applicable for mining qualified web traversal patterns than the two sequential mining algorithms, ISL and IncSpan. Therefore, in this paper, the goal of our algorithm is the same as IncWTP which is mining qualified web traversal patterns in the environment where user sequences can be inserted into or deleted from the web traversal sequence database. Although IncWTP use extended lattice structure to store most information of web traversal sequence database so that it can mine qualified web traversal patterns incrementally, the structure is too huge to be hold in memory. Hence it should be saved in hard disk. during performing the incremental mining process, each level in the lattice will be loaded in memory and maintained by turns. But this process will take very much time for I/O. Besides, IncWTP has to scan the original database when new candidates are generated during performing the incremental mining process.

### IV. IncWTP_PLM Algorithm

#### A. Initial step: constructing projected-database link matrix and page frequency table

A page frequency table (PFT) that stores information about pages in the web traversal sequence database. Each entry in the page frequency table has two fields: page and count. Page and count store the name of the page and the count of that link in the web traversal sequence database, respectively. If there are \( m \) pages, the size of table is \( m \) as shown in TABLE II.

<table>
<thead>
<tr>
<th>page</th>
<th>( p_1 )</th>
<th>( p_2 )</th>
<th>( \ldots )</th>
<th>( p_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>( C_1 )</td>
<td>( C_2 )</td>
<td>( \ldots )</td>
<td>( C_m )</td>
</tr>
</tbody>
</table>

A projected-database link matrix (PLM) is built from the web traversal sequence database according to the qualified web traversal patterns, where predecessor page as rows and successor page as columns (The row and column of PLM represent the predecessor page and successor page, respectively). For each entry \(( i, j)\) in the matrix records two kinds of information, the count of web traversal sequences which visited the successor page \( p_j \) after the predecessor page \( p_i \) and the links of \( \langle p_i, p_j \rangle \)-projected-database. For each entry in PLM is not null if and only if there exists an out-link from the predecessor page to the successor page in the website structure. Suppose our web site is very huge. In the web site, the number of page is 10,000 and the average number of out-link for each page is 20. The storage requirement of \( \text{projected-database link matrix} \) is just 200,000 integers. In other words, it costs memory most 1Mb to store \( \text{projected-database link matrix} \). If there are \( m \) pages, the dimension of matrix is \( m \times m \) as shown in TABLE III.

<table>
<thead>
<tr>
<th>predecessor</th>
<th>( p_1 )</th>
<th>( p_2 )</th>
<th>( \ldots )</th>
<th>( p_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td>successor</td>
<td>( \langle C_{i_1}, L_{i_1} \rangle )</td>
<td>( \langle C_{i_2}, L_{i_2} \rangle )</td>
<td>( \ldots )</td>
<td>( \langle C_{i_m}, L_{i_m} \rangle )</td>
</tr>
</tbody>
</table>

### TABLE II. Page Frequency Table

### TABLE III. Projected-database Link Matrix

\(^1\)In this paper, we do not present the extended lattice structure because of the space limitation.
spatial matrix. Without storing the entries which is set as null, the storage space can be saved.

For instance, we consider the web traversal sequence database as TABLE I. It stores a set of users’ sessions which has two fields: Sid stores the session id and Sequence stores the sequence of page which is ordered by traversal time. Given such input web traversal sequence database, the algorithm of PLM construction construct PFT, PLM and the projected-databases with the prefix of length 2 at the same time as follows:

1) The algorithm first traverses the input graph (website structure) to create an 9 by 9 projected-database link matrix PLM_{9x9} and a page frequency table PFT. Immediately, all entries in the PFT are set as 0. In addition, the count part of these entries in the PLM_{9x9}, which are the out-link from the predecessor page to the successor page in the website structure, are set as 0.

2) The algorithm scans the input database once (web traversal sequence database), and then obtains the PFT and the PLM_{9x9} which are as shown in TABLE IV. and Figure 2, respectively. At the same time, <pipj>-projected-database [6][7] are produced, if the C_{ij} of the entry (i, j) of PLM_{9x9}, is greater than 0. For example, since C_{1,2} = 7 is greater than 0, <ab>-projected-database is generated as shown in Figure 2. Besides, we will make file link in the entry (1, 2) to point the <ab>-projected-database. Finally, we can determine the number of sequences, tol, in the input database when we finish the scan of it. In this example, tol would be determined as 10.

B. Mining Qualified Web Traversal Patterns

In this subsection, we present an algorithm which utilizes the outputs from the step of PLM construction (i.e. tol, PFT, PLM and the projected-databases with frequent prefix of length 2) and minimum support ms to mine qualified web traversal patterns. First, we determine the minimum support count by production of tol and ms. As above example, tol = 10 and ms is set to 40%, the minimum support count msc should be determined as 10*40% = 4. Then we scan PFT to identify those pages whose support count greater than or equal to msc as qualified web traversal patterns with length 1. Consider the PFT as TABLE IV., the patterns <a>, <b>, <c>, <d>, <e>, <g> and <i> must be found while qualified web traversal patterns with length 1.

Finally, the PLM is scanned for identifying qualified web traversal patterns with length 2. For each qualified web traversal patterns with length 2 <pipj>, we use the link in PLM to address and load the <pipj>-projected-database in memory from hard disk. Because web traversal sequence database should be a sparse dataset, most projected-databases with prefix of length 2 should be small. Therefore, projected-databases with frequent prefix of length 2 could be load in memory.

Although there probably exist a few projected-databases with frequent prefix of length 2 whose size is bigger than capacity of memory, we could utilize batch processing or Pseudo Projection in WTPrefix-Span Algorithm. And then given such input <pipj>-projected-database and the graph (website structure) use the algorithm WTPrefix-Span for recursively mining the qualified web traversal patterns with length greater than 2 as follows:

1) Constructing a-PFT step. We first traverse the input graph (website structure) to find the successor pages of p_i for creating a local page frequency table a-PFT, where p_i is the last page of the prefix a of the input the projected-database.

2) Identifying qualified web traversal patterns step. We then perform one scan of the input database (a-projected-database) to obtain support counts for entries of a-PFT.

3) Producing local projected database step. We then perform a second scan of the input database (a-projected-database) to obtain the local projected database (a'-projected-database). The prefix of database is the qualified web traversal patterns produce by identifying qualified web traversal patterns step. Recursively, we recursively call the WTPrefix-Span algorithm for each local projected database Produced by producing local projected database step.

C. Incremental Step: Mining Qualified Web Traversal Patterns Incrementally

TABLE V. INSERTED WEB TRAVERSAL SEQUENCE DATABASE

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sequence</th>
<th>Sid</th>
<th>Sequence</th>
<th>Sid</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>bgiehca</td>
<td>12</td>
<td>acebg</td>
<td>13</td>
<td>dcdbg</td>
</tr>
</tbody>
</table>

TABLE VI. DELETED WEB TRAVERSAL SEQUENCE DATABASE

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>biabehi</td>
</tr>
<tr>
<td>2</td>
<td>bgabe</td>
</tr>
</tbody>
</table>

In this subsection, the proposed incremental algorithm, called IncWTP.PLM, utilizes the outputs from the step of
PLM construction (i.e., tol, PFT, PLM and the projected-databases with frequent prefix of length 2) and minimum support ms to mine new qualified web traversal patterns from incremental web traversal sequence database. Actually, the difference between IncWTP, PLM and the algorithm of mining qualified web traversal patterns in subsection B is that IncWTP, PLM maintains tol, PFT, PLM and the projected-databases with frequent prefix of length 2 by one scan of the incremental web traversal sequence database.

As we known, an incremental web traversal sequence database consists of two parts: the inserted web traversal sequence database (as shown in TABLE V.) and the deleted web traversal sequence database (as shown in TABLE VI.). For instance, consider the an incremental web traversal sequence database (as shown in TABLE VI.).

For instance, consider the an incremental web traversal sequence database as TABLE V. and TABLE VI. and outputs from the step of PLM construction. Given such input incremental web traversal sequence database and data structure, the process of discovering new traversal patterns is stated below.

1) Insertion Step. We perform one scan of the inserted web traversal sequence database as shown in TABLE V. to maintain support counts for entries of PFTs and count part of entries of PLM_{i,j}. At the same time, if the C_{ij} of the (i, j) entry of PLMs_{i,j} is increased, the L_{ij} of the (i, j) entry of PLMs_{i,j} will be marked and <p_{ij}>-projected-database will be updated. For example, since C_{1,2} is increased, <ab>-projected-database is updated. Also we can determine the number of sequences, tol, in the input database when we finish scanning. In this instance, tol should be determined as 13.

![Projected-database link matrix example](image)

Figure 3. An example of projected-database link matrix in deletion step

2) Deletion Step. We perform one scan of the deleted web traversal sequences database as shown in TABLE VI. to maintain counts for PFTs and count part of PLM_{i,j} as shown in TABLE VII. and Figure 3., respectively. At the same time, we mark the L_{ij} of PLM_{i,j} to be decreased, and store the Sid of deleted web traversal sequences database in a list delList. Then, after scanning the deleted web traversal sequence database, the tol should be determined as 11.

3) Re-discovery Step. After scanning the inserted and deleted web traversal sequences database, <p_{ij}>-projected-database is loaded in memory, if C_{ij} is marked. In the loaded <p_{ij}>-projected-database, the sequence whose Sid is contained in delList is deleted. Then we save the new <p_{ij}>-projected-database in hard disk and further consider it as input for calling WTPrefix-Span algorithm.

![Projected-databases with the prefix of length two](image)

TABLE VII. PAGE FREQUENCY TABLE EXAMPLE

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>11</td>
<td>10</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>

V. EXPERIMENTAL EVALUATION

In this section, we present experimental results on the real data, KDD Cup 2000 dataset. Since there is only one incremental mining algorithm, IncWTP [3], for mining qualified web traversal patterns currently, we use it to compare with our algorithm IncWTP, PLM.

The KDD Cup 2000 dataset can be downloaded from the URL: http://cobweb.ecn.purdue.edu/KDDCUP/. The dataset is contains 777,480 clicks, 234,954 sessions and 137 different Request Templates. We treat Request Template as page and group the dataset by session id. In each group, the pages are order by referred time as session sequence. Thus we have 234,954 session sequences and 137 different pages. The average length of the sequences is 3.287. We divide the dataset into two parts: one with size 188K as original user session database and another with size 47K as the inserted dataset. Then, deleted datasets were generated by sampling from the user sessions with amount varied from 4.7k to 47K. The most frequent out link is contained in 27,806 sessions. That is the biggest projected-databases with prefix of length 2 contain at most 27,806 sessions. In average, the length of the sequences is 3.287. We divide the dataset into two parts: one with size 188K as original user session database and another with size 47K as the inserted dataset. Then, deleted datasets were generated by sampling from the user sessions with amount varied from 4.7k to 47K. The most frequent out link is contained in 27,806 sessions. That is the biggest projected-databases with prefix of length 2 contain at most 27,806 sessions. In average, the length of the sequences is 3.287. We divide the dataset into two parts: one with size 188K as original user session database and another with size 47K as the inserted dataset. Then, deleted datasets were generated by sampling from the user sessions with amount varied from 4.7k to 47K. The most frequent out link is contained in 27,806 sessions. That is the biggest projected-databases with prefix of length 2 contain at most 27,806 sessions. In average, the length of the sequences is 3.287. We divide the dataset into two parts: one with size 188K as original user session database and another with size 47K as the inserted dataset. Then, deleted datasets were generated by sampling from the user sessions with amount varied from 4.7k to 47K. The most frequent out link is contained in 27,806 sessions.

![Scalability with size of deleted dataset](image)

Figure 4. Scalability with size of deleted dataset

Figure 4. and Figure 5. show that our algorithm and IncWTP, scale linearly to the size of incremental datasets (i.e. inserted datasets and deleted datasets) in incremental step. Our algorithm tends to outperform the IncWTP. However, our algorithm does not outperform IncWTP significantly when minimum support is set as 2% and incremental dataset is of small size. The reason is that most levels of the
extended lattice structure can be fit in memory. Therefore, the I/O cost is significantly less than the case that minimum support set as 1% or 0.8%.

**Figure 5.** Scalability with size of inserted dataset

**Figure 6.** Scalability with minimum support of deleted dataset

**Figure 7.** Scalability with minimum support of inserted dataset

VI. COPYRIGHT FORMS AND REPRINT ORDERS

In this paper, we proposed an efficient incremental data mining algorithm named *IncWTP* for mining qualified web traversal patterns when the new user sequences are inserted into and old user sequences are deleted from original database. We proposed some new data structures, *PFT* and *PLM*, to store the useful information to avoid re-mining the original database. Hence, the qualified web traversal patterns can be discovered efficiently when the traversal sequence database is updated. The Experimental results show that *IncWTP* algorithm is about 3–5 times faster than *IncWTP* algorithm under numbers of experiment where the minimum support is set as 0.8%.

For future work, we shall investigate another interesting issue that the time-gap between two pages in a session sequence is important for navigation prediction. Hence, we shall also investigate discovering time-gap qualified traversal patterns incrementally.

**ACKNOWLEDGMENT**

This research was supported by National Science Council, Taiwan, R.O.C. under grant no. NSC 97-2631-H-006-001, and by Ministry of Economic Affairs, Taiwan, R.O.C. under grant no. 95-EC-17-A-02-51-024

**REFERENCES**


