A Weighted Graph Web Usage Mining Method to Evaluate Usage of Websites

1Mehdi Heydari, 2Raed Alsaqour, 1Khairil Imran, 1Kamelia Vaziry

1Department of Computer Science, Faculty of Information Technology, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia.
2Department of Computer Science, Faculty of Information Science and Technology, University Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia.

Abstract: Web Usage Mining (WUM) is a method to evaluate usage of websites. Traditionally, WUM uses a Server Log File (SLF) as a web usage data source. SLF is confronted with three problems: (i) in some cases such as page cashing or pushing of the back button of the browser, no data is recorded in SLF. It causes a considerable amount of web usage data to be lost and consequently the accuracy of WUM is decreased. (ii) In the linear web browsing model, the sequence of pages visited corresponds to the sequence of SLF records whereas in the parallel web browsing model the sequence of pages visited does not correspond to the sequence of SLF records and thus does not match with actual web browsing. WUM methods are therefore faced with difficulty during the reconstruction of the user web browsing model. (iii) Sometimes, it is discovered that web usage patterns are not essentially interesting patterns because of site structure. So a method is needed to distinguish web usage patterns during patterns discovery. To cope with the abovementioned problems, this paper proposes a Weighted Graph WUM (WGWUM) method, which consists of an AJAX interface and a Custom Log File (CLF). The AJAX interface monitors the events of all data source levels and records them into the CLF. To cope with the parallel web browsing model, a graph mining algorithm is applied, which helps users to define a threshold value to determine which patterns are valuable. To evaluate the proposed WGWUM method, a robot software was designed to simulate user web browsing behavior. The robot is able to navigate through websites and records all of their activities. In addition, the robot randomly determines the page that should be visited and the duration of page visiting. When its web browsing finishes its navigation on the website, the graph mining algorithm is applied on SLF and CLF files. WGWUM method shows 100% accuracy on discovering the traversed paths whereas through existing methods it is 73%. This accuracy helps web administrators to improve their websites especially when the time is a significant factor, such as in e-learning web-based systems.

Key words: Web usage mining, page browsing time, graph mining algorithm.

INTRODUCTION

With the growth and rapid multiplication of web-based systems, the volumes of Web Usage Data (WUD) collected by web servers have reached huge proportions. Analyzing such data can help website owners to optimize the functionality, content, and structure of their websites. WUM is a method to analyze WUD. WUM applies data mining methods on WUD to discover web usage patterns. Item-set mining, Subsequence mining, and Graph mining are three kinds of data mining methods (Mihara et al., 2007). WUM consists of three steps: preprocessing, pattern discovery, and pattern analysis. WUM can be collected from three sources, at: Server level, Client level and Proxy level (Jaideep et al., 2000). WUM uses SLF that is a server level data source. Depending on the configuration of the web server, the SLF comes in various formats. Basically, it has two formats: a common log format and an extended common log format. The common log format includes the following five fields (Zdravko and Daniel, 2007). Remote host field, Date/Time field, HTTP request field, status code field, and transfer volume field. The extended common log format is a common log format with two additional fields, the referrer field and the user agent field. A referrer is the URL of a previous item which led to the web request. For each HTTP request, one record, containing request details, is appended to the end of the SLF.

The graph mining method that is applied on WUD is called Graph Based Web Usage Mining (GWUM). GWUM that is applied on a weighted graph is called Weighted Graph Web Usage Mining (WGWUM). WGWUM perceives the website structure as a vertex weighted graph where each vertex represents a web page, each edge represents the link between web pages, and the vertex’s weight represents the numerical value assigned to the web page. The vertex’s weight is used to distinguish between vertices. This study proposes a WGWUM method that covers all WUD sources and takes page browsing time into account. In addition, this paper proposes a users’ browsing behavior analysis approach which is based on applying web usage mining...
techniques. The proposed users’ browsing behavior analysis approach is beneficial for the area of website design improvement.

The Study of WUM was started in 1996. Chen et al., in (1998) presented the first WUM work. The authors applied datamining on web usage data to find hidden Webknowledge. In addition, the authors clarified the data preparation stage of web usage mining, relying to a great extent on the correct reconstruction of a user’s path. The methods in prevalent use today rely on the temporal order of clicks, by viewing a session as a single clickstream. Chen et al. have shown the limits of such a linear order when the user navigates with multiple parallel windows or tabs. The clickstream simply does not reflect the user’s path if only the temporal order is considered. We have developed the WGWUM method, which takes such parallel browsing behavior into account and use it to develop a tree structure approach encompassing all paths a user could have taken. A tree structure contains all paths a user might have taken and uses it to reconstruct session data. The case study has shown the necessity of taking parallel browsing behavior into account. Any web usage analysis not accounting for parallel browsing behavior will not produce accurate results.

Several works have been accomplished since that time. In (Suneetha and krishnamoorthi, 2009) discussed various sources of Web Log Files (WLFs) such as web server log, proxy log and client log. The authors discussed the structure of WLF in detail and performed two preprocessing techniques: data cleaning and user identification. However, performing one or two techniques cannot guarantee reliable results through WUM. Sessions identification is another very important technique at preprocessing level, which the authors did not apply.

In Murata and Saito (2006) collected the users’ accesses from the modified client web log. A user search keyword and graph are generated from it. Thereafter, users’ interests are also mined from the graph by applying the PageRank algorithm to assign importance to accessed pages. In the next step, unimportant nodes and weak edges were removed from the graph. In the last phase, the graph is decomposed into further sub-graphs, which depict the surfing behavior of users. Some sort of cleaning was performed by removing unimportant nodes and noisy data from the log data but no other significant technique was used for more precise preprocessing. Users’ interests can be mined in a better way by grouping the interests based on pages visited in a particular time interval.

In Hirate and Yamana (2006) proposed the sequential pattern mining method which extracts patterns taking into account the interval of the time each item occurred, by calculating and discretizing the difference from the time the first item occurred. This approach treats the time as a distance of items, and in the case applying to WUM, it is suited to evaluate the achievement of the conversion on marketing.

**MATERIALS AND METHODS**

The problem statement in this study is divided into three sub-problems: (i) Incomplete web usage data, (ii) Complexity of web browsing path, and (iii) Evaluation of discovered patterns. To make the problem statement more understandable, some definitions are presented first.

During web browsing, the user has five options: (1) to open the page in the current window, (2) to open a page in a new window, (3) to open a page in new tab, (4) to switch between tabs or windows, or (5) to move to visited pages by clicking the back or forward button of the web browser.

**Definition:**

We define the Open function with three parameters to present user’s web browsing behavior as follows:

**Open (Referrer, Target, Type)**

Here Referrer is the current webpage which is being browsed by the user. For the first request, where there is no opened page, Referrer is empty (Φ). Target is the page which is requested through Referrer. Type is one of the five web browsing options: CurrentWindow, NewWindow, NewTab, Switch and Return.

**Incomplete Web Usage Data:**

SLF may not be fully reliable during reconstruction of a user’s session because in some cases such as page caching, post method, or pushing of the back button of browser, web browsing data is not recorded in SLF (Jaideep et al., 2000), so a hole appears in the SLF data. This hole is usually filled in the path completion step of the preprocessing phase. However, in some cases, path completion cannot be useful. One example is that of an unknown referrer (when referrer of webpage is unknown); the unknown referrer cannot be solved by path completion. The following example describes this problem.
**Example 1:**
Suppose a user follows this scenario: Opens page A in a new window, goes through page A, opens page B in a new tab, switches to page A and opens page C in a new tab, switches to page B and opens page D in the current window, switches to page C, opens page D in the current window and opens page E in current window. Such user browsing behavior can be represented by the following functions:

1) Open (Φ, A, NewWindow);
2) Open (A, B, NewTab);
3) Open (B, A, Switch);
4) Open (A, C, NewTab);
5) Open (C, B, Switch);
6) Open (B, D, CurrentWindow);
7) Open (D, C, Switch);
8) Open (C, D, CurrentWindow);
9) Open (D, E, CurrentWindow);

Figure 1(a) illustrates the web browsing model. Figure 1(b) shows the page request data in SLF. When we want to reconstruct the user’s navigation paths based on data in SLF, we cannot determine which D is a referrer of E (Figure 2(a)). In other words, the navigation path is not clear (Figure 2(b)).

![Web browsing model](image)

<table>
<thead>
<tr>
<th>Referrer</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Φ</td>
<td>A</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>B</td>
<td>D</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>D</td>
<td>E</td>
</tr>
</tbody>
</table>

**Fig. 1:** (a) Web browsing model, (b) Server log file.

![Unknown referrer](image)

**Fig. 2:** (a) Unknown referrer, (b) Unknown traversal path.

**Complexity of Web Browsing Path:**
In linear web browsing, where the user opens one webpage and at the same time follows links, the sequence of recorded data on SLF corresponds to the user’s session. But in parallel web browsing where the user opens several pages in a new window or tab at the same time and navigates between them, the sequence of recorded data on SLF does not correspond to the user’s session. Therefore, among data mining methods, Graph Mining is the most appropriate choice to manage this case and data mining methods such as sequential pattern mining are not applicable anymore.

**Example 2:**
In linear web browsing, the sequence of recorded data in SLF corresponds to the sequence of the real navigation path, so the user’s session can be reconstructed using SLF data. Following example shows a linear web browsing data.

1) Open (Φ, A, NewWindow);
2) Open (A, B, NewTab);
3) Open (B, C, NewTab);
4) Open (C, D, CurrentWindow);
Figures 3(a), 3(b), and (3c), represent sample SLF data, sequence of page request based on SLF data, and web browsing model respectively.

<table>
<thead>
<tr>
<th>Referrer</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Φ</td>
<td>A</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

(a) \( \rightarrow \) B, B \( \rightarrow \) C, C \( \rightarrow \) D

(b) \( \rightarrow \)

(c) \( \rightarrow \)

Fig. 3: (a) Server log file, (b) Sequence of page request based on SLF data, (c) web browsing model.

But in parallel web browsing, the sequence of SLF does not correspond to the sequence of the real navigation path, so the user’s session cannot be reconstructed using SLF data. To illustrate the problem, suppose the user follows this scenario: Opens page A in a new window, goes through page A, opens page B in new tab, switches to page A, goes through page A, opens page C in a new tab, switches to page B, then goes from page B go to page D. Such user browsing behavior can be represented by the following functions:

1) \textit{Open (Φ, A, NewWindow)};
2) \textit{Open (A, B, NewTab)};
3) \textit{Open (B, A, Switch)};
4) \textit{Open (A, C, NewTab)};
5) \textit{Open (C, B, Switch)};
6) \textit{Open (B, D, CurrentWindow)};

Figures 4(a), 4(b), 5(a) and (5b) represent sample SLF data, sequence of page request based on SLF data, web browsing model based on SLF, and web browsing model based on real web browsing respectively.

<table>
<thead>
<tr>
<th>Referrer</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Φ</td>
<td>A</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>B</td>
<td>D</td>
</tr>
</tbody>
</table>

(a) \( \rightarrow \) B, A \( \rightarrow \) C, B \( \rightarrow \) D

(b) \( \rightarrow \)

Fig. 4: (a) Server log file, (b) Sequence of page request based on SLF data.

Fig. 5: (a) Web browsing model based on SLF, (b) Web browsing model based on real web browsing.

**Evaluation of Discovered Patterns:**

The discovered web usage pattern does not essentially express the importance of it. For example, suppose we find the sequence \(<A,C>\) as a frequent traversal path whereas because of website structure users have to visit pages \(A\) and \(C\) to reach to pages \(E\) and \(F\). In fact, users are interested in page \(E\) or \(F\). Therefore, we need to find a way to evaluate discovered patterns. Patterns can be evaluated after the pattern discovery process. However, instead of discovering irrelevant patterns and then throwing them out, it is better to apply a method, which can do a primary evaluation during the mining process to minimize finding irrelevant patterns. Giving weight to items can be considered as a good method in which the weight of each item represents its importance. As we have mentioned, graph mining can be used to manage complex web browsing behavior. Assigning weight to graph vertices (or web pages) helps us to define a boundary for patterns that are mined. Browsing time is a
suitable variable that can be used for weighting. For example, if browsing time of a webpage is less than what is expected, it is a sign that the webpage content is not useful or the website structure is in a form that causes the page to be visited before reaching the target page.

**Exactness of Browsing Time:**

When items are given weight, the data mining algorithm relies on these weights to discover patterns. Therefore, it is very important to calculate and assign weights accurately. As we mentioned before, assigning browsing time to web pages is an example of weighting. Sometimes because of web browsing features such as those arising from the difference between client side data and server side data, misevaluation of page browsing time calculation occurs. For example, recalling parallel web browsing where users open several pages at the same time, in such cases data recorded in SLF only shows the requested time of the pages and cannot help us to find out which page and for how long has been really browsed on the client machine.

**Definitions:**

- **Session Start** ($S_J$): is the time at which user session is started and it is the time of the first user request.
- **Session End** ($S_E$): is the time at which the user session expires and occurs when the user does not send any request to the web server for certain duration of time.
- **Request Time** ($R_X$): is the time of webpage request.
- **Browsing Time** ($B_X$): is the time duration of webpage visit.

**Examples:**

For example, consider the following scenario: the user opens page $A$ in a new window, from page $A$, opens page $B$ in a new tab, then from page $B$, switches to page $A$, and from page $A$, opens page $C$ in a new tab. Such user behavior can be represented by the following functions:

1. Open ($\Phi$, $A$, NewWindow);
2. Open ($A$, $B$, NewTab);
3. Open ($B$, $A$, Switch);
4. Open ($A$, $C$, NewTab);

Figures 6(a), 6(b), 6(c) illustrate the Browsing model, Server log file data, and Browsing time of web pages based on SLF respectively.

**Example 4:**

Based on example 3, now suppose the user switches to page $B$, just two seconds after opening page $C$ according to the following user behavior functions:

1. Open ($\Phi$, $A$, NewWindow);
2. Open ($A$, $B$, NewTab);
3. Open ($B$, $A$, Switch);
4. Open ($A$, $C$, NewTab);
5. Open ($C$, $B$, Switch);

Figures 7(a), 7(b), and 7(c) illustrate the Web browsing model, Server log file data, and Browsing time of web pages based on SLF respectively.
As it can be considered, in both examples (examples 3 and 4) the data recorded in the server log file are same. However, there is a difference between real browsing times (figure 6(c) and 7(c)) when the browsing time is calculated based on SLF data.

RESULTS AND DISCUSSION

Architecture of the Proposed WGWUM Method:

Figure 8 presents architecture of the proposed WGWUM method. An AJAX interface is placed between the browser and web server to monitor and record the user’s web browsing behaviors; it includes requests that are sent to the server and activities that are done on the client side such as switching between tabs or windows.

Fig. 8: Proposed WGWUM Architecture.

Phases of the Proposed Method:

The proposed WGWUM method consists of three phases: (i) Data Collection, (ii) Data Preprocessing, and (iii) Pattern Discovery. Figure 9 shows the phases of the proposed WGWUM method.

Fig. 9: Phases of the proposed WGWUM method.

Phase 1: Data Collection:

This phase is related to the monitoring and recording of the user’s web browsing behaviors. To record the user’s web browsing behaviors, we use remote agent technology. To do this, we designed an AJAX interface which helps us to obtain both client and server side data. It controls five events: (1) On Session Start, (2) On Session End, (3) On Page Request, (4) On Page Load, and (5) On Page Focus. On Session Start, On Session End and On Page Request are server side events while On Page Load and On Page Focus are client side events.

On Session Start:

This event occurs when user requests a web page from the web server for the first time. When this event arises, a unique ID, SessionID, is generated. SessionID helps to segregate user sessions from each other during session reconstruction.

Algorithm 1, On Session Start:

```
//generate session ID
SessionID = GenerateSessionID();
```

On Page Request:

This event arises every time the user requests a webpage from the web server. When this occurs, a unique ID, PageID, is generated and assigned to the requested page (Ps).

Algorithm 2, On Page Request:

```
//Generate Unique ID
```
PageID = GeneratePageID();
//Assign a unique ID (PageID) to requested page (P)
AssignPageID(P, PageID);
//Register script to monitor client side browsing behaviors
RegisterClientScript();
}

On Page Load:
This is a client side event that occurs when a web page is loaded on the user’s browser. When this event arises, the PageID of Referrer, PageID of Target, Browsing Time, Event Name, Date, and Time are recorded in session. Browsing time is left blank until the preprocessing phase.

Algorithm 3, On Page Load
{
    //Write information in session
    WriteToSession (SessionID, TargetPageID, ReferrerPageID, BrowsingTime, Event, Date, Time);
}

On Page Focus:
This is a client side event that occurs when the user focuses on a web page. When this event arises, some data similar to the OnPageLoad event is recorded in a session.

Algorithm 4, On Page Focus:
{
    //Write information in session
    WriteToSession (SessionID, TargetPageID, ReferrerPageID, BrowsingTime, Event, Date, Time);
}

On Session End:
This event occurs when the user doesn’t send any request to the web server for a predefined duration of time which is called the Session Timeout. When this event arises, the content of the user session is recorded in a file named CLF on the server by web application.

Algorithm 5, On Session End:
{
    //Write session data into custom log file
    For each record R in session
    {
        WriteToLogFile(R);
    }
}

The Web browsing data are recorded in the user’s session until the session is expired. Then, all data is transmitted to the CLF. CLF is similar to SLF where unused fields have been eliminated and instead some useful fields have been added. It consists of eight fields: Session ID, Page Name, Page ID, Referrer ID, Browsing Time, Event, Date, and Time.

Definitions:
- Session ID: identifier of the current session
- Page Name: name of page
- Page ID: ID of page
- Referrer Id: identifier of referrer of page
- Browsing Time: browsing time of page
- Event: type of page, onFocus or onLoad
- Date: date of requesting page
- Time: time of requesting page

Phase 2: Data Preprocessing:
This phase is related to the converting of the CLF data to the graph structure. In this phase, web usage data is converted to the graph structure in a way which can be used for the graph mining method. In this phase, a base graph is constructed, the users’ session is separated, the user’s traversal path is discovered, a traversal...
database is constructed, and browsing time of each page is calculated and assigned to a corresponding vertex in the base graph.

**Algorithm 6, Data Preprocess:**

\[
\begin{align*}
&\text{for each SessionID (SID) in CLF} \\
&\quad \text{Traversal } T = \text{FindTraversal}(SID); \\
&\quad \text{for each Page } P \text{ in } T \\
&\quad \quad \text{RemoveClientID}(P); \\
&\quad \quad \text{AppendToTraversalDataBase}(SID, T); \\
&\quad \quad \text{for each Page } P \text{ in } T \\
&\quad \quad \quad \text{BrowsingTime (BT) = CalculateBrowsingTime}(P); \\
&\quad \quad \quad \text{AssignBrowsingTime}(P, BT); \\
&\quad \end{align*}
\]

**Phase 3: Pattern Discovery:**

This phase is related to applying the graph mining method to discover web usage patterns. The Seongand Hyu graph mining method in (Seong and Gyu, 2009) is applied to discover weighted frequent patterns.

**Algorithm 7, Mining Weighted Frequent Patterns:**

\[
\begin{align*}
&\text{MineWeightedFrequentPattern}(\text{BaseGraph}, \text{TraversalDB}, \text{MBT}); \\
&\end{align*}
\]

To clarify the proposed WGWUM method, we present an example. Suppose a user navigates using the following user’s behavior functions:

1) Open (Φ, A1, NewWindow);
2) Open (A1, B1, NewWindow);
3) Open (B1, A1, Switch);
4) Open (A1, C1, NewWindow);
5) Open (C1, E1, NewWindow);
6) Open (E1, C1, Switch);
7) Open (C1, F1, NewWindow);
8) Open (F1, E1, Switch);
9) Open (E1, G1, CurrentWindow);
10) Open (G1, B1, Switch);
11) Open (B1, D1, NewWindow);
12) Open (D1, B1, Switch);
13) Open (B1, C2, NewWindow);
14) Open (C2, F2, NewWindow);
15) Open (F2, C2, Switch);
16) Open (C2, E2, CurrentWindow);
17) Open (E2, D1, Switch);
18) Open (D1, G2, CurrentWindow);

The user’s behavior in functions (1-18) is represented by the web browsing model as shown in figure 10. Here 3 phases of the proposed WGWUM method are explained.
Fig. 10: Web browsing model.

**Phase 1 (Data Collection):**
All browsing behaviors are monitored and kept in the user session until session timeout. After session timeout, data is recorded in CLF as shown in Table 1.

<table>
<thead>
<tr>
<th>SID</th>
<th>P</th>
<th>PID</th>
<th>RID</th>
<th>BT</th>
<th>E</th>
<th>D</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>A</td>
<td>A1</td>
<td>-</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T1</td>
</tr>
<tr>
<td>01</td>
<td>A</td>
<td>A1</td>
<td>-</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T2</td>
</tr>
<tr>
<td>01</td>
<td>B</td>
<td>B1</td>
<td>A1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T3</td>
</tr>
<tr>
<td>01</td>
<td>B</td>
<td>B1</td>
<td>A1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T4</td>
</tr>
<tr>
<td>01</td>
<td>A</td>
<td>A1</td>
<td>B1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T5</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C1</td>
<td>A1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T6</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C1</td>
<td>A1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T7</td>
</tr>
<tr>
<td>01</td>
<td>E</td>
<td>E1</td>
<td>C1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T8</td>
</tr>
<tr>
<td>01</td>
<td>E</td>
<td>E1</td>
<td>C1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T9</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C1</td>
<td>E1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T10</td>
</tr>
<tr>
<td>01</td>
<td>F</td>
<td>F1</td>
<td>C1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T11</td>
</tr>
<tr>
<td>01</td>
<td>F</td>
<td>F1</td>
<td>C1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T12</td>
</tr>
<tr>
<td>01</td>
<td>E</td>
<td>E1</td>
<td>F1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T13</td>
</tr>
<tr>
<td>01</td>
<td>G</td>
<td>G1</td>
<td>E1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T14</td>
</tr>
<tr>
<td>01</td>
<td>G</td>
<td>G1</td>
<td>E1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T15</td>
</tr>
<tr>
<td>01</td>
<td>B</td>
<td>B1</td>
<td>G1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T16</td>
</tr>
<tr>
<td>01</td>
<td>D</td>
<td>D1</td>
<td>B1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T17</td>
</tr>
<tr>
<td>01</td>
<td>D</td>
<td>D1</td>
<td>B1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T18</td>
</tr>
<tr>
<td>01</td>
<td>B</td>
<td>B1</td>
<td>D1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T19</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C2</td>
<td>B1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T20</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C2</td>
<td>B1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T21</td>
</tr>
<tr>
<td>01</td>
<td>F</td>
<td>F2</td>
<td>C2</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T22</td>
</tr>
<tr>
<td>01</td>
<td>F</td>
<td>F2</td>
<td>C2</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T23</td>
</tr>
<tr>
<td>01</td>
<td>C</td>
<td>C2</td>
<td>F2</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T24</td>
</tr>
<tr>
<td>01</td>
<td>E</td>
<td>E2</td>
<td>C2</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T25</td>
</tr>
<tr>
<td>01</td>
<td>E</td>
<td>E2</td>
<td>C2</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T26</td>
</tr>
<tr>
<td>01</td>
<td>D</td>
<td>D1</td>
<td>E2</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T27</td>
</tr>
<tr>
<td>01</td>
<td>G</td>
<td>G2</td>
<td>D1</td>
<td>0</td>
<td>L</td>
<td>D1</td>
<td>T28</td>
</tr>
<tr>
<td>01</td>
<td>G</td>
<td>G2</td>
<td>D1</td>
<td>0</td>
<td>F</td>
<td>D1</td>
<td>T29</td>
</tr>
</tbody>
</table>

**Phase 2 (Data Preprocessing):**
In this phase, using CLF, the traversal graph is constructed as shown in figure 11(a). Based on the traversal graph, all paths are discovered, ClientIDs are removed, and the traversal database is constructed as shown in figure 11(b).

Using the website structure, we construct the base graph as shown in figure 12(a). Then the total browsing time of each page is calculated and is assigned to the corresponding vertex in the base graph as shown in figure 12(b).

![Traversal graph](image1)

![Traversal database](image2)

Fig. 11: (a) Traversal graph, (b) Traversal database.
Phase 3 (Pattern Discovery):

Now that the traverse graph and traversal database are ready, giving a minimum browsing time, we can apply the graph mining method in (Seong and Hue, 2007) to discover Weighted Frequent (WF) patterns.

Tables 2, 3 and 4 illustrate the results of applying the weighted frequent patterns mining algorithm in (Seong and Gyu, 2007) with minimum weight support 6, pattern length k=1, k=2, k=3. Here P, W, SC, SB, S, WS, WB, WF, and F are respectively abbreviations of Patterns, Weight, SCount, SBound, Support, WSsupport, WBound, Weighted Frequent, and Feasible. The details of the definition and calculation of the above terms can be found in (Seong and Hyu, 2007).

Based on the results of tables 2, 3, and 4, the discovered weighted frequent patterns are: \{C, CE, CF\} as shown in tables 2 and 3. These patterns represent the output of the proposed WGWUM method.

Figure 13 presents experimental results of the proposed WGWUM method. The figure illustrates the number of discovered frequent web usage patterns. The minimum weight support or minimum browsing time varies from 1 minute to 29 minutes, and since the session time-out of most websites is 30 minutes, this range was chosen. As can be observed, the number of discovered patterns in the case of server side data is more than when both server and client side data are used. Using the WGWUM method, extra discovered patterns are junk patterns that have been discovered because of wrong data; it means that in some cases there is no accurate data (for example due to an unknown referrer) is produced using statistics and probability, which decreases the accuracy of discovered patterns.

Fig. 12: (a) Base graph, (b) Base Graph with browsing time.

Fig. 13: Experimental result, discovered frequent web usage patterns (minwsup: 1 to 29).
In the next experiment using a robot, traversal paths discovery are done two times, once on SLF and again on CLF. Here the number of session is 100. Figure 14 represents the accuracy of the discovered traversal path in both cases with and without considering client side data.

**Fig. 14:** Experimental result, the accuracy of discovered traversal paths.

**Conclusion:**

This paper has proposed a novel WGWUM method to discover web usage patterns. Apart from previous WUM methods, we considered the website as a graph and user navigation path as traversals on it. Then we applied the graph mining method in (Seong and Hue, 2007) to discover web usage patterns. The WGWUM method is able to filter discovered patterns based on their importance. It is done by assigning weights (browsing time) to vertices (web page).

These patterns help administrators to evaluate the current usage of their website. In this paper, client side events such as page focus and page load were used to monitor user activities on the client side. As a future work, it is suggested to consider more events such as page scrolling to find out if the page is really being used or had just has been left open.

**REFERENCES**


