Capturing the Semantics of Web Log Data by Navigation Matrices

Wilfred Ng
Email:wilfred@cs.ust.hk
Department of Computer Science, The Hong Kong University of Science and Technology

Abstract: The information left behind by users who have visited a web site is recorded in the related web server log files. From analysing the data contained in such files, a web designer is able to understand the interaction between the users and a web site, and then to improve the web topology. We assume that the information of web usage can be generated from log files via a cleaning process, from which we identify a set of navigation sessions that represent the trails formed by users during the navigation process. The trails are modelled as a weighted directed graph, called a transition graph, and then a corresponding navigation matrix is computed with respect to the underlying web topology. The main contribution in this paper is that we formally define a minimal set of binary operators on navigation matrices, which consists of the sum, union, intersection and difference operators. These operations afford us the ability to analyse users navigation from the contents of two given navigation matrices.

Key words: Web Navigation, Web Log Data, Trails, Navigation Matrices

1. INTRODUCTION

A web site is one of the most important components pertaining to the infrastructure for running different modes of Electronic Commerce (EC) such as B2B (or B2C) [Shaw99]. In order to assist users in searching for or purchasing products and services, a web site designed for EC purpose is usually rich in information content and is complicated in hyperlink structure. Thus, it is essential that the site topology should be well-designed for presenting information to potential customers or relevant business partners. Due to fierce business competition in the EC field, it is also important to understand clearly the navigation behaviour of the

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35658-7_21
R. Meersman et al. (eds.), *Semantic Issues in E-Commerce Systems*

users in cyberspace so as to evolve the information contents and the hyperlink structures of a web site.

The interaction between a web site and the users can be found in the related web server log files, which record a large amount of data concerning the navigation details of when and how the users visit the web server [CMS99, Mena99]. There have been many studies concerning the use of adapted data mining techniques on web log data [ZXM98, CPY98, PE00, PMZ00, NC01]. However, most of them return only statistical information such as page counts or navigation patterns governed by the conventional parameters of confidence and support in such analysis (see [PE00] and [BL98] for example). We adopt a matrix-theoretic approach in modelling web log data and propose a set of algebraic operators, collectively called *navigation operators*, which can be employed to manipulate navigation matrices. We now show the basic concept of how to use navigation matrices for analysing log data in the block diagrams given in Figure 1.

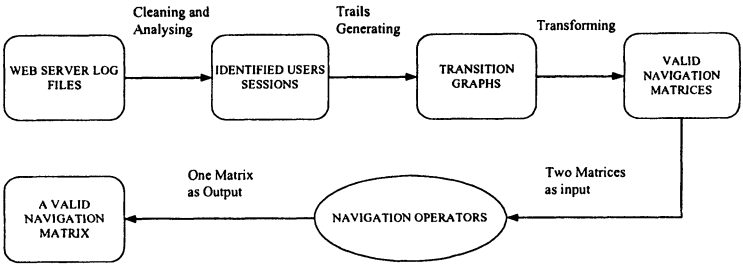


Figure 1. The block diagram showing the process of web log data analysis

A web server log file contains the raw log data of the usage details. It needs to be cleaned before human readable and machine processable. The mechanism used in the process also depends on the web log data format. For more details of cleaning raw log data, the readers may refer to [CMS99]. In this work, we stay indifferent to the methods used in cleaning as long as they are consistent and the user navigation sessions can be identified in the output of the cleaning process. We model the log information of a web site over a web topology as a *directed graph* (or simply a *digraph*) [BKM+00], in which a node represents a web page and a link represents a transition via a hyperlink from one web page to another.

We also need to assume that the pages recorded in a user session are closely related. A user session can then be viewed as a *trail* (i.e. a sequence of visited pages) generated within a reasonable time period over the web topology that represents the structure of a web site. We first extend the web topology by including a source page and a finishing page, in addition to the pages that exist in the site. Then a weighed digraph, called a *transition graph*, is generated by superimposing all identified sessions on the underlying web topology. In other words, the weight of a link represents the number of times in traversing the links. Consequently, we transform a transition graph into a *navigation log data matrix* (or simply a *navigation matrix*),

which is served as a fundamental notion to study the user navigation behaviour. We define a set of sound navigation operators on navigation matrices, which is briefly described as the following table.

Operators	Brief Descriptions
Sum (+)	To add up the trails that are inferred from two log files over the same web topology.
Union ($\cup$)	To overlap the trails that are inferred from two log files over the same web topology.
Difference (−)	To minus the trails that are inferred from one log file from another over the same web topology.
Intersection ($\cap$)	To obtain the common trails that are inferred from two log files over the same web topology.

Table 1. Brief description of navigation operators

The use of the above operators in analysing log data is desirable, since they are easy to understand and to compute, and their output results are ready to present as a transition graph. More importantly, the intuition of *overall*, *change* and *common* web site usage can be formalised by the operations as highlighted in Table 1.

2. GENERATING NAVIGATION MATRICES FROM WEB SERVER LOG DATA

There are two sources of log files: (1) server log files and (2) personal log files (e.g. browsing history of a user in a proxy). For convenience in discussion, we refer to the log data obtained from the first source, though we note that the navigation operations defined later on can be applicable to that from the second source as well. The data recorded in log files (possibly more than one) reflects the (possibly concurrent) access of a web site by multiple users such as the domain name (or the IP address) of the request, the user who generated the request (if applicable) and the URL of the referring page. The log data can be stored in various formats in a log file, for example, NCSA Common Log Format [CMS99] as shown in Figure 2.

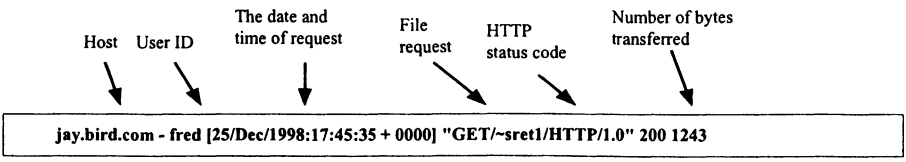


Figure 2. A log entry in NCSA Common Log Format

The log data can be employed to reconstruct the user navigation sessions within the site. We need to generate a human readable and machine processable form of log data via the process of data cleaning, in which the useful log entries are identified as an output. As an HTML page may contain linkages to image, sound, or video files, the corresponding file transfer protocols used on the web are required to establish a separate connection for each file requested. In a cleaning process the log entry of the HTML file is the only entry corresponding to a file explicitly requested by the user, and all other log entries such as those requesting *gif* or *pdf* files can be ignored.

We view a web site as a network of linked web pages (or simply linked pages) together with an interface that allows a user to browse the contents of pages. We call the network topology of a web site *the web topology*. A user enters into a web site is allowed to get access the pages in a non-sequential way defined by the underlying web topology. We now formalise this idea in Definition 2.1.

Definition 2.1 (Web Topology) A *web topology* **W** is an ordered pair (**P**, **L**) where **P** is a set of n web pages = $\{P_1, \ldots, P_n\}$ for some positive integer n, and **L** is a binary relation on **P** = $\{(P_i, P_j) \mid P_i, P_j$ are two distinct pages in **P** with $1 \leq i, j \leq n\}$, which represents a set of hyperlinks between pages contained in **P**.

A web topology **W** can be naturally perceived as a *directed graph* (or a digraph) in which no loop is allowed for any single page. It is also easy to see that a user should follow the links defined in **L** when visiting a web site. Let us further illustrate this idea by the example given in Figure 3, which shows a digraph representation of a web topology with **P** = $\{P_1, P_2, P_3\}$ and **L** = $\{(P_1, P_2), (P_2, P_1), (P_2, P_3), (P_1, P_3)\}$.

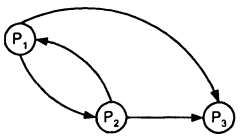


Figure 3. An example of a web topology W viewed as a digraph

Given **W** = (**P**, **L**), a *trail* T over **W** is a non-empty sequence of pages in **P** such that every pair of consecutive pages in T is a link in **L**. A page P_j is said to be *reachable* from another page P_i over **W** if there exists a possible trail from P_i to P_j, or else P_j is said to be *not reachable* from P_i. Trivially, if $(P_i, P_j) \in$ **L**, then P_j is reachable from P_i. (But the converse of this statement is not true.) Formally, given two distinct pages $P_i, P_j \in$ **P**, P_j is reachable from P_i if and only if (P_i, P_j) is in the *transitive closure* of **L**. The transitive closure of **L** is a binary relation, denoted by $\mathbf{L}^+$ and is defined as $\{(P_A, P_B) \mid$ there exists $k > 0$ and $(P_{Ai}, P_{Bi}) \in$ **L** for $1 \leq i \leq k$-1 such that $P_{A1} = P_A$, $P_{Bi} = P_{A(i+1)}$ and $P_{Bk} = P_B\}$.

Example 1 We can easily see in Figure 3 that P_2 is reachable from P_1, and P_3 is reachable from P_2. However, P_2 is not reachable from P_3. A collection of trails defined on the topology **W** given in Figure 3 is shown in Figure 4.

Trails
$P_2 \rightarrow P_1 \rightarrow P_2 \rightarrow P_3$
$P_1 \rightarrow P_2 \rightarrow P_3$
$P_1 \rightarrow P_2$
$P_2 \rightarrow P_1$

Figure 4. A possible set of trails over W

We view user interaction within a web site **W** as a collection of user navigation sessions whose information is embedded in log files. A user navigation session being inferred from a log file is modelled as a trail, which represents a sequence of requests made by the user within a defined time interval. In an ideal scenario each user is allocated a unique IP address when accessing a web site. We assume that a user visits the site more than once, each time possibly with a different goal in hand. A user session is therefore defined as a sequence of requests from the same IP address such that no two consecutive requests are separated by more than X minutes, where X is a given parameter. In [CP95] the authors report an interesting finding that 25.5 minutes is a reasonable time interval between requests within a user session. The following simplified table illustrates the inferred sessions from log data.

IP Address	URL Requested		Time of the Request	
123.456.78.9	B.html	(P2)	2000/04/02-10:25:10	$X = 25.5$ minutes
123.456.78.9	A.html	(P1)	2000/04/02-10:29:11	
123.456.78.9	B.html	(P2)	2000/04/02-10:29:43	*Session 1*
123.456.78.9	C.html	(P3)	2000/04/02-10:30:27	
123.456.78.9	A.html	(P1)	2000/04/02-11:30:28	
123.456.78.9	B.html	(P2)	2000/04/02-11:30:43	*Session 2*
123.456.78.9	C.html	(P3)	2000/04/02-11:30:57	
123.456.78.10	A.html	(P1)	2000/04/02-11:50:24	
123.456.78.10	B.html	(P2)	2000/04/02-11:56:06	*Session 3*
123.456.78.11	B.html	(P2)	2000/04/02-11:56:16	
123.456.78.11	A.html	(P1)	2000/04/02-11:57:13	*Session 4*

Figure 5. User sessions inferred from cleaned log data

We make two further assumptions in our study. First, there is a *starting page* S and a *finishing page* F, in addition to those pages contained in **P**. The inclusion of these two pages are necessary, since it is reasonable to expect a user can enter into a page in **P** from some external pages, or leave from a page in **P** to some external pages in practice. Second, a collection of user navigation sessions, which represent a set of trails **T**, can be obtained in a data cleaning process as the output result. Specifically, the following usage information, $|SP_i|$, $|P_iF|$ and $|P_iP_j|$ from **T** with $1 \leq i, j \leq n$, are computed by Algorithm 1 given below. The semantics of $|SP_i|$, $|P_iF|$ and

$|P_iP_j|$ are the weights of the links from S to P_i, from P_i to F, and from P_i to P_j, respectively.

Algorithm 1 (a set of trails **T** defined over **W** having n pages)
ASSIGN each page in **W** an index running from $i = 1$ to n;
FOR $1 \leq i, j \leq n$, DO $|SP_i| = |P_iF| = |P_iP_j| = 0$;
FOR $i = 1$ to n DO
$|SP_i|$ = The number of times that a page P_i was first requested (i.e. P_i being the first page in a trail in **T**);
$|P_iF|$ = The number of times that a page P_i was last requested (i.e. P_i being the last page in a trail in **T**);
FOR $j = 1$ to n and $j \neq i$ DO
$|P_iP_j|$ = The number of times that two pages P_i and P_j appearing as consecutive pages in a trail in **T**;
END FOR;
END FOR;
RETURN $|SP_i|$, $|P_iF|$ and $|P_iP_j|$ for $i, j \in \{1, \ldots, n\}$

We incorporate the values of $|SP_i|$, $|P_iF|$ and $|P_iP_j|$ where $i, j \in \{1, \ldots, n\}$ into a given web topology **W** to generate a corresponding weighted digraph $\mathbf{G_w}$, which we call *a transition graph*. A transition graph is constructed from **W** by including the two additional pages S and F. Moreover, the weight of each link is determined by $|SP_i|$, $|P_iF|$ or $|P_iP_j|$ accordingly. We do not show in $\mathbf{G_w}$ any link that has zero weight (i.e. a link never being traversed). The diagram given in Figure 6 shows a transition graph which corresponds to the topology **W** given in Figure 3 and the set of trails given in Figure 4.

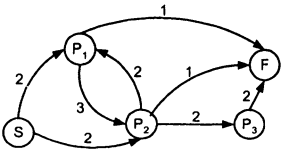


Figure 6. A transition graph representing user sessions over a web topology

We observe that, given a weighted digraph over **W** with S and F, it may not necessarily entail a "correct" transition graph. In other words, it may be the case that there does not exist any set of trails **T** over **W** such that by using Algorithm 1 the given weighted digraph can be generated. So we need the concepts given in Definition 2.2 in order to ensure that a given $\mathbf{G_w}$ is a *valid* transition graph.

Definition 2.2 (Balanced Page, Page Degree and Valid Transition Graph) A page P_i in **W** is said to be *balanced* if the total weight of its in-links is equal to the total weight of its out-links in a transition graph. We call the sum of the weights of in-links and out-links of a page P_i the *degree* of P_i. A weighted digraph $\mathbf{G_w}$

represents a transition graph defined over **W** is said to be a *valid* transition graph if, for all $i, j \in \{1, \ldots, n\}$, it satisfies the four conditions given as follows: (1) The weights of the links from S to S, F to F, S to F, P_i to S and F to P_j are zero; (2) Every link from P_i to P_j having non-zero weight is also a link in **W** (i.e. $(P_i, P_j) \in \mathbf{L}$). (Note that this excludes looping in a page.); (3) Every P_i in $\mathbf{G_w}$ should be balanced; and (4) Every P_i in $\mathbf{G_w}$ which has non-zero degree should be reachable from S.

The first condition is to characterise the special pages S and F in order to allow users enter into and leave from a page in a web site as discussed. The second condition asserts that any trail should be supported by the underlying web topology. The third and fourth conditions characterise the fact that the weights of links are formed by superimposing all the trails in the digraph. Clearly, a transition graph obtained from Algorithm 1 which takes a set of user sessions inferred from a log file as input should be valid. We now give the definition of a navigation matrix.

Definition 2.3 (Navigation Matrix) A *navigation matrix* over **W**, denoted as $\mathbf{M_w}$, is a square matrix $[a_{ij}]$ with dimension n+2, where *by convention* the entry a_{ij} represents the element in the ith row and the jth column with $1 \le i, j \le n$+2. We let P_0 = S and P_{n+1} = F in order to have a uniform notation. The value of a_{ij} is defined to be the weight associated to the link from $P_{(i-1)}$ to $P_{(j-1)}$ (i.e. $|P_{(i-1)}\, P_{(j-1)}|$ in $\mathbf{G_w}$). A navigation matrix is said to be valid if its corresponding transition graph is valid.

A navigation matrix is a useful tool that defines for analysing web log data. We now present an algorithm to compute a navigation matrix $\mathbf{M_w}$ from a given transition graph $\mathbf{G_w}$ representing user navigation sessions inferred from a log file.

Algorithm 2 ($\mathbf{G_w}$ with pages $\{S, F\} \cup \{P_1, \ldots, P_n\}$)
DECLARE a square matrix $\mathbf{M_w}$ with dimension n+2;
FOR all $1 \le i, j \le n$+2, DO $a_{ij} = 0$;
FOR all $1 < i, j < n$+2 and $i \neq j$, DO $a_{ij} = |P_{(i-1)}P_{(j-1)}|$;
FOR all $1 < i, j < n$+2, DO
$\quad a_{1j} = |SP_{(j-1)}|$; $a_{i(n+2)} = |FP_{(i-1)}|$;
END FOR;
RETURN $\mathbf{M_w}$

Trivially, there is a one-to-one correspondence between the class of valid transition graphs and the class of valid navigation matrices. It is also clear that Algorithm 2 returns a valid navigation matrix $\mathbf{M_w}$, since $\mathbf{G_w}$ satisfies the criterion stated in Definition 2.2. In Figure 7, we show a (valid) navigation matrix $\mathbf{M_w}$ with 3 pages P_1, P_2 and P_3 in **W**, corresponding to $\mathbf{G_w}$ given in Figure 6, for example $a_{12} = |SP_1| = 2$, $a_{34} = |P_2P_3| = 2$, and $a_{43} = |P_3P_2| = 0$. The indicative templates of pages are added in $\mathbf{M_w}$ for the sake of easy referencing.

$$\mathbf{M_w} = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 & 1 \\ 0 & 2 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Figure 7. The navigation matrix $\mathbf{M_w}$ corresponding to $\mathbf{G_w}$

From now on, we utilise the terms transition graphs and navigation matrices interchangeably due to the conceptual duality. We present the following proposition, which essentially adapts the criterion of a valid transition graph in the context of navigation matrices. Informally, the first part shows that our idea of using a navigation matrix to represent a transition graph is correct. The second part shows that the accessed pages should start and finish properly. The third and fourth parts are due to the first and second conditions of Definition 2.2.

Proposition 2.1 Let $\mathbf{M_w}$ be a navigation matrix with dimension n+2. Then the following statements are true.

1. A navigation matrix is an unambiguous representation of a navigation graph.
2. Given a non-zero entry in ith row or jth column in $\mathbf{M_w}$ with $1 < i, j < n+2$. Then the pages P_i or P_j are on some trails in the corresponding transition graph $\mathbf{G_w}$ such that their starting page is S and finishing page is F.
3. The first column, the last row and the diagonal running from a_{11} to $a_{(n+2)(n+2)}$ of a given navigation matrix contain only zero entries.
4. $a_{1(n+2)} = 0$. □

3. THE NAVIGATION MATRIX OPERATORS

In this section we define the four operators of the sum, the union, the difference, and the intersection, each of which takes two given navigation matrices over a web topology as input parameters, and returns a navigation matrix representing a valid transition graph as an answer. The output result provides a deeper insight for a web designer to check if the topology can achieve the expected web usage.

We define the *in-degree* of a page $P_{(i-1)}$ to be $\sum_{k=1}^{n+2} a_{ki}$ and the *out-degree* of a page to be $\sum_{k=1}^{n+2} a_{ik}$, where $1 \le i \le n+2$. (Recall that we assume P_0 = S and P_{n+1} = F.) We also denote the binary operators *min* and *max* the usual minimum and maximum of two given integers. We use throughout this subsection M_1 and M_2 to represent two navigation matrices defined over the same web topology **W**. The sum operation is now given in Definition 3.1, which adds up the two input navigation matrices and returns a navigation matrix as an output to represent the overall web usage.

Definition 3.1 (Sum of Navigation Matrices) The *sum* of two navigation matrices M_1 and M_2, denoted as $M_1 + M_2$, is defined as a navigation matrix M_3 over **W** such that for all $i, j \in \{1, \ldots, n+2\}$, $(a_{ij})_3 = (a_{ij})_1 + (a_{ij})_2$.

An interesting property of the sum operation is that the sum of two navigation matrices M_1 and M_2, which originates from the two respective navigation graphs G_1 and G_2 inferred from the log files F_1 and F_2, is equal to the matrix which originates from the navigation graph G_3 inferred from the log file integrating the information in both F_1 and F_2. The informal reason is that the number of traversals and the in-degree and out-degree are preserved under the sum operation.

Theorem 3.1 Let M_1, M_2 and M_3 be the navigation matrices corresponding to the navigation graphs G_1, G_2 and G_3 inferred from the log files F_1, F_2 and F_3, respectively, where F_3 is the file obtained from merging the files F_1 and F_2. Then $M_3 = M_1 + M_2$.

(Proof Outline.) The theorem can be established by using Definition 3.1 and induction on the number of pages in M_1 and M_2 respectively.

Theorem 3.1 is significant since it implies that we are able to perform analysis on overall navigation information by summing up individual pieces of navigation information, in this sense we say that the sum operator is *additive* with respect to the log data. Another reasonable way to analyse the overall navigation behaviour is to consider the maximum number of traversals in a link, which represents the coverage of the links in a web topology, taking account of the overlap of traversals.

Definition 3.2 (Union of Navigation Matrices) The *union* of two navigation matrices M_1 and M_2, denoted as $M_1 \cup M_2$, is defined as a navigation matrix M_3 over **W** such that for all $i, j \in \{2, \ldots, n+1\}$,

1. $(a_{ij})_3 = max((a_{ij})_1, (a_{ij})_2)$;
2. $(a_{1j})_3 = max((a_{1j})_1, (a_{1j})_2) + max(0, \sum_{k=1}^{n+2} max((a_{jk})_1, (a_{jk})_2) - \sum_{k=1}^{n+2} max((a_{kj})_1, (a_{kj})_2))$;
3. $(a_{i(n+2)})_3 = max((a_{i(n+2)})_1, (a_{i(n+2)})_2) + max(0, \sum_{k=1}^{n+2} max((a_{ki})_1, (a_{ki})_2) - \sum_{k=1}^{n+2} max((a_{ik})_1, (a_{ik})_2))$; and
4. all other elements are zero.

Note that in Definition 3.2 both the equations (2) and (3) take account of the difference between the total weights of in-coming and out-going links of a page, as defined in their second *max* term. This is in order to ensure that all pages except P_0 and $P_{(n+2)}$ are balanced in the output answer M_3, a necessary condition to be a valid matrix. Also, the second term in both the equations (2) and (3) is necessarily to be non-negative because we allow only one link direction in the special pages S and F. We remark that the equation (4) fills all entries of the first column and last row with

zero values (c.f. parts (3) and (4) of Proposition 2.1). Similar remarks can also apply to the operators in Definitions 3.3 and 3.4 given later on.

We now introduce the difference operator, which is useful to compare the difference in contents of two web log files. For example, we can use the difference operator to compare the temporal change in two sets of log data obtained at different time intervals or to compare the log data obtained from two user groups having different profiles related to some business objectives.

Definition 3.3 (Difference of Navigation Matrices) The *difference* of two navigation matrices M_1 and M_2, denoted as $M_1 - M_2$, is defined as a navigation matrix M_3 over **W** such that for all $i, j \in \{2, \ldots, n+1\}$,

1. $(a_{ij})_3 = max(0, ((a_{ij})_1 - (a_{ij})_2))$;
2. $(a_{1j})_3 = max(b_0, ((a_{1j})_1 - (a_{1j})_2)) +$
   $max(0, \sum_{k=1}^{n+2} max(b_1, ((a_{jk})_1 - (a_{jk})_2)) - \sum_{k=1}^{n+2} max(b_2, ((a_{kj})_1 - (a_{kj})_2))$,
   where $b_0 = 1$ if $(a_{1j})_1 \neq 0$, or else $b_0 = 0$, $b_1 = 1$ if $(a_{jk})_1 \neq 0$, or else $b_1 = 0$, and $b_2 = 1$ if $(a_{kj})_1 \neq 0$, or else $b_2 = 0$;
3. $(a_{i(n+2)})_3 = max(b_0, (a_{i(n+2)})_1 - (a_{i(n+2)})_2)) +$
   $max(0, \sum_{k=1}^{n+2} max(b_1, ((a_{ki})_1 - (a_{ki})_2)) - \sum_{k=1}^{n+2} max(b_2, ((a_{ik})_1 - (a_{ik})_2))$,
   where $b_0 = 1$ if $(a_{i(n+2)})_1 \neq 0$, or else $b_0 = 0$, $b_1 = 1$ if $(a_{ki})_1 \neq 0$, or else $b_1 = 0$, and $b_2 = 1$ if $(a_{ik})_1 \neq 0$, or else $b_2 = 0$; and
4. all other elements are zero.

Similar to Definition 3.2, the second *max* term in the equations (2) and (3) are used to balance the total weights of in-links and out-links in the output answer M_3. It appears to be more straightforward to define these equations in a *stricter* form by putting $b_0 = b_1 = b_2 = 0$ (i.e. in this case the definition becomes subtracting the weight of links in M_1 from that in M_2 in a trivial way). However, such a stricter definition may lead to the problem of an isolated set of pages (i.e. partitioning the corresponding transition graph into two separate portions), which violates the fourth condition in Definition 2.2, and leads to an invalid navigation matrix as an output answer. The following example illustrates this problem when putting $b_1 = b_2 = 0$.

Example 2 In this example we use the two transition graphs G_1 and G_2 in Figures 8(a) and 8(b) to represent the navigation matrices M_1 and M_2 respectively. We then have the *invalid* graph in Figure 8(c), if we do not differentiate the cases of b_1 and b_2 when determining a_{1j} and $a_{i(n+2)}$ in the second and third equations of Definition 3.3. On the other hand, the output result of our definition produces a valid navigation matrix corresponding to the transition graph given in Figure 8(d).

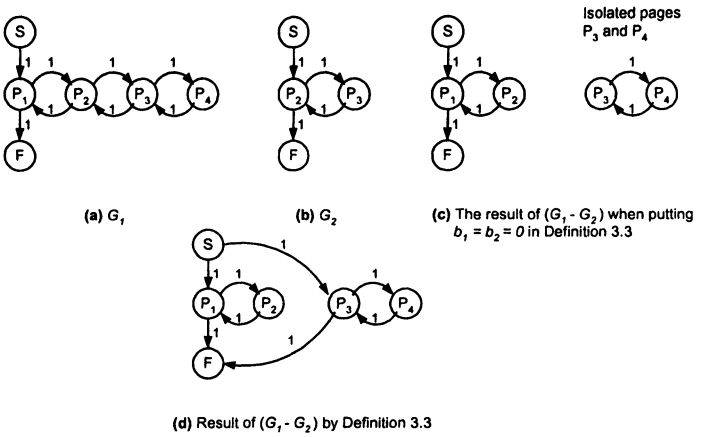


Figure 8. Defining difference in a trivial way may lead to problems

Definition 3.4 (Intersection of Navigation Matrices) The *intersection* of two navigation matrices M_1 and M_2, denoted as $M_1 \cap M_2$, is defined as a navigation matrix M_3 over **W** such that for all $i, j \in \{2, \ldots, n+1\}$,

1. $(a_{ij})_3 = max(0, min((a_{ij})_1, (a_{ij})_2))$;
2. $(a_{1j})_3 = max(b_0, min((a_{1j})_1, (a_{1j})_2)) +$
   $max(0, \sum_{k=1}^{n+2} (max(b_1, min((a_{jk})_1, (a_{jk})_2)) - \sum_{k=1}^{n+2} (max(b_2, min((a_{kj})_1, (a_{kj})_2)))$,
   where $b_0 = 1$ if $(a_{1j})_1 \neq 0$, or else $b_0 = 0$, $b_1 = 1$ if $(a_{jk})_1 \neq 0$, or else $b_1 = 0$, and $b_2 = 1$ if $(a_{kj})_1 \neq 0$, or else $b_2 = 0$; and
3. $(a_{i(n+2)})_3 = max(b_0, min((a_{i(n+2)})_1, (a_{i(n+2)})_2)) +$
   $max(0, \sum_{k=1}^{n+2} (max(b_1, min((a_{ki})_1, (a_{ki})_2)) - \sum_{k=1}^{n+2} (max(b_2, min((a_{ik})_1, a_{ik})_2)))$,
   where $b_0 = 1$ if $(a_{i(n+2)})_1 \neq 0$, or else $b_0 = 0$, $b_1 = 1$ if $(a_{ki})_1 \neq 0$, or else $b_1 = 0$, and $b_2 = 1$ if $(a_{ik})_1 \neq 0$, or else $b_2 = 0$.

The reader may find that in Definition 3.4 we also differentiate the cases for b_0, b_1 and b_2. The reason is similar to Definition 3.3, that is, assuming $b_0 = b_1 = b_2 = 0$ may lead to the problem of an isolated set of pages. The following example helps to illustrate this problem of putting $b_0 = 0$.

Example 3 In this example we use the two transition graphs G_1 and G_2 in Figures 9(a) and 9(b) to represent the navigation matrices M_1 and M_2 respectively. We can see that the invalid transition graph is resulted as shown in Figure 9(c). On the other hand, the output result of our definition produces a valid navigation matrix corresponding to a valid transition graph as shown in Figure 9(d).

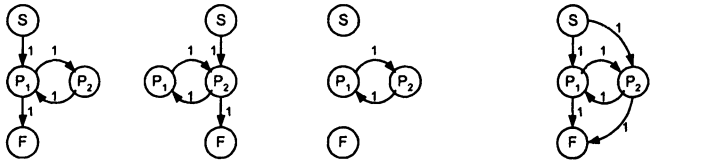


Figure 9. Defining intersection in a trivial way may lead to problems

We also observe that, in general, all operations except the difference are *commutative*, which is now stated as follows.

Proposition 3.1 $M_1 \theta M_2 = M_2 \theta M_1$ where θ is the sum, union or intersection operators. □

We remarks that the four navigation operations from Definitions 3.1 to 3.4 generate a valid navigation matrix as the output answer M_3, in this sense the four operators are said to be *sound*. Generally speaking, all these operations (1) ensure the pages from P_1 to P_n are balanced, (2) prevent the formation of isolated set of pages, and (3) preserve the link direction in S and F pages.

Theorem 3.2 The navigation operations in Definitions 3.1 to 3.4 are sound.
(Proof Outline.) It is easy to show by induction on the number of the pages that the output answers (i.e. M_3) obtained from Definitions 3.1 to 3.4 are valid matrices defined over **W**, since their corresponding transition graphs satisfy the four conditions given in Definition 2.2.

4. AN EXAMPLE FOR NAVIGATION OPERATIONS

In this section, we make use the following two transition graphs G_1 and G_2 given in Figure 10, which are assumed to be defined over the same web topology, to illustrate the use of the four operations in more detail.

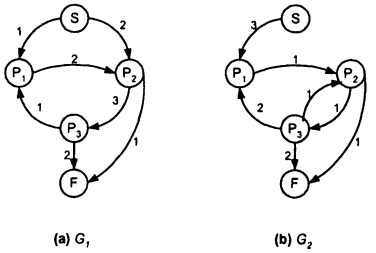


Figure 10. Two transition graphs used for illustration

The navigation matrices M_1 and M_2 corresponding to G_1 and G_2 are given as follows:

$$M_1 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad M_2 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The diagram given in Figure 11 is the result of $M_3 = M_1 + M_2$ and the corresponding transition graph.

$$M_3 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 4 & 2 & 0 & 0 \\ 0 & 0 & 3 & 2 & 0 \\ 0 & 0 & 0 & 4 & 2 \\ 0 & 1 & 1 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad G_3 =$$

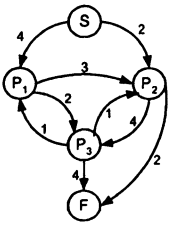


Figure 11. Result of $M_3 = M_1 + M_2$

The following is the result of $M_4 = M_1 \cup M_2$ and the corresponding transition graph. Note that $|P_2F|$ in G_4 has become 2 instead of 1 as shown in G_1 and G_2. This is in order to keep the page P_2 balanced, as discussed in Definition 3.2.

$$M_4 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 3 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 0 & 3 & 2 \\ 0 & 1 & 1 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad G_4 =$$

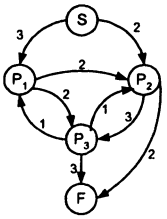
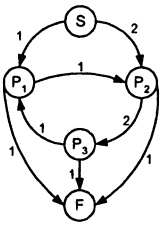


Figure 12. Result of $M_4 = M_1 \cup M_2$

The diagram given in Figure 13 is the result of $M_5 = M_1 - M_2$ and $M_6 = M_2 - M_1$ and the corresponding graphs G_5 and G_6.

$$M_5 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad G_5 =$$



$$M_6 = \begin{array}{c} \\ S \\ P_1 \\ P_2 \\ P_3 \\ F \end{array} \begin{array}{c} \begin{array}{ccccc} S & P_1 & P_2 & P_3 & F \end{array} \\ \begin{bmatrix} 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \qquad G_6 =$$

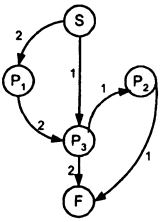


Figure 13. Results of $M_5 = M_1 - M_2$ and $M_6 = M_2 - M_1$

The diagram given in Figure 14 is the result of $M_7 = M_1 \cap M_2$ and the corresponding transition graph.

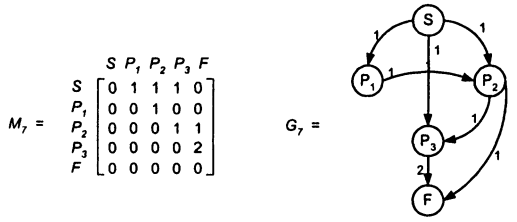


Figure 14. Result of $M_7 = M_1 \cap M_2$

Finally, we discuss the idea of applying the navigation operators in analysing the derivation between a designer's anticipation that concerns the usage of a web topology and the actual usage inferred from the web log data. First, the designer's expectation can be formalised and represented as a valid navigation matrix, denoted by M_{exp}. This can be obtained by imposing the web designer's belief of the number of traversals on each hyperlink of the related web topology. The belief may take account of the resource allocation and content distribution. Another way to construct M_{exp} is to use statistical methods to simulate the weight of links in a web topology (c.f. [ZL99]). This approach requires further establishment of a statistical model on site usage, which can be founded on the empirical data obtained from experiments on real site users. The discrepancy between the web designer's expectation and the users' behaviour can be formalised by using the sum and the union of all log data. The former is to compute the difference between M_e and $(\sum_{i=1}^{n} M_i)$ by $(M_e - \sum_{i=1}^{n} M_i)$, where $(\sum_{i=1}^{n} M_i)$ represents the sum of n related navigation matrices that are derived from a set of log files $\{F_1, \ldots, F_n\}$ corresponding to the server(s) of a web site. The latter is to compute the difference between M_e and $(\bigcup_{i=1}^{n} M_i)$ by $(M_e - \bigcup_{i=1}^{n} M_i)$, where $(\bigcup_{i=1}^{n} M_i)$ represents the union of all navigation matrices that are derived from the log files. Based on the information obtained from the output matrix, or equivalently the corresponding transition graph, the designer can better visualise the deviation between the expected usage and the actual usage, since $(\sum_{i=1}^{n} M_i)$ and $(\bigcup_{i=1}^{n} M_i)$ formalise the idea of *mass customisation* of all the site users. The resource allocation can also be referenced to the result of $(\bigcap_{i=1}^{n} M_i)$, since $(\bigcap_{i=1}^{n} M_i)$ formalises the idea of the most popular set of trails inferred from the collected data.

5. CONCLUDING REMARKS

In this paper we studied a collection of user sessions that are inferred from server log files defined over a web topology formalised in Definition 2.1. The user sessions are perceived as a valid transition graph as in Definition 2.2, whose information can be modelled as a navigation matrix as in Definition 2.3. We formally defined a set of binary operations from Definitions 3.1 to 3.4, which includes the sum, the union, the

difference and the intersection, on using navigation matrices as input parameters. The operations enhance the capabilities of analysing site usage. We also showed that these four operations are sound in Theorem 3.2, in the sense that they always output a valid navigation matrix as an answer, in which all pages that have non-zero state in the corresponding transition graph are reachable and balanced. The sum operation is shown to be additive with respect to log data in Theorem 3.1. We are currently incorporating these operations into our web log analysis system, which supports a web designer's decision in reconstructing a web site in order to adapt the navigation needs of users. As navigation matrices are usually sparse in practice (i.e. only small percentage of matrix entries are non-zero), we have to study some special storage and programming techniques that are useful to deal with the sparseness in order to minimise the storage costs and computation time. There are also limitations on using web server log files to infer navigation sessions, due to the fact that cache and proxy servers are commonly used in a web configuration.

Acknowledgement: This work is supported by the Hong Kong Polytechnic University Grant POLYU5095/00E. The author would like to thank all anonymous referees for their constructive comments.

REFERENCES:

[BL98] J. Borges and M. Levene. Mining association rules in hypertext databases. In *Proc. of the 4th Int. Conf. on Knowledge Discovery and Data Mining*, pp. 149-153, (1998).

[BKM+00] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins and J. Wiener. Graph structure in the web. In *Proc. of the 9th WWW Conf.*, (2000).

[CPY98] M.S. Chen, J. S. Park and P. S. Yu. Efficient data mining for traversal patterns. *IEEE Transactions on Knowledge and Data Engineering*, 10(2) pp. 209-221, (1998).

[CMS99] R. Cooley, B. Mobasher and J. Srivastava. Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1) pp. 5-32, (1999).

[CP95] L. D. Catledge, and J. E. Pitkow. Characterizing browsing strategies in the world wide web. Computer Networks and ISDN Systems, 27(6) pp. 1065-1073, (1995).

[Mena99] J. Mena. *Data mining your website.* Digital Press, (1999).

[Ng99] W. Ng. Evaluating the client side approach and the server side approach to the WWW and DBMSs integration. In *Proc. of the 9th Int. Database Workshop,* pp. 72-82, (1999).

[NC01] W. Ng. and C. Chan. WHAT: A web hypertext associated trail mining system. In *Proc. of the 9th IFIP 2.6 Working Conf. on Database Semantics*, pp. 205-220, (2001).

[PMZ00] J. Pei, J. Han, B. Mortazavi-asl and H. Zhu. Mining access patterns efficiently from web logs. In *Proc. of PAKDD Conf.*, pp. 396-407, Japan. (2000).

[PE00] M. Perkowitz and O. Etzioni. *Towards adaptive web sites: conceptual framework and case study. Artificial Intelligence*, 118(2000) pp. 245-275. (2000).

[Shaw99] M. Shaw. *Handbook on electronic commerce.* Springer-Verlag, (1999).

[SF98] M. Spiliopoulou and L. Faulstich. WUM: a tool for web utilization analysis. In *Proc. of the International Workshop on the Web and Databases*, pp. 184-203, (1998).

[ZL99] N. Zin and M. Levene. Constructing web-views from automated navigation sessions. In *Proc. of the ACM Digital Lib. Workshop on Organizing Web Space*, pp. 54-58, (1999).