Effectively Capturing User Navigation Paths in the Web Using Web Server Logs

Amithalal Caldera and Yogesh Deshpande

School of Computing and Information Technology, College of Science Technology and Engineering, University of Western Sydney
PO Box 1797, Penrith South DC, NSW 1797, Australia
{h.caldera,y.deshpande}@uws.edu.au

Abstract. Most of the approaches to analyse the Web server logs to capture user access patterns are heuristic based and affected by the use of proxy servers, caching and stateless service model of the HTTP protocol. No heuristic has addressed all of these problems. In this paper, we propose a new heuristic to overcome this limitation. The heuristic exploits the background knowledge of user navigational behaviour recorded in the server logs without requiring additional information through cookies, logins and session ids. The heuristic is evaluated by analysing the logs of a university Web server that records user ids for administrative reasons, which allows us to compare it against the concrete knowledge of user sessions. We also evaluate our heuristic against some of the existing heuristics. The evaluation has shown very satisfactory result.

1 Introduction

A Web server log explicitly records the browsing behaviour of site visitors and consists of details about file requests to a Web server and the server responses to those requests. A typical Web server log contains information such as the IP address of the machine that made the request, the date and time a request was made, the request method that the client used (GET, POST), the protocol used, the URL of the requested page, the status code of the response message, the size of the document transferred, the URL of the referrer page from which the request was initiated and the user agent, which is the application software used to browse the Web.

A sequence of requests to a server from a single user within a certain time window is called *a user session*. The powerful user tracking techniques such as requiring authentication [9], storing cookies [4] or generating session IDs [7] have been used in capturing the user sessions on the Web. To protect the privacy of users, Web server logs in general do not record such information unless an application, such as in e-commerce, requires it. In the absence of explicit knowledge of user identities, most of the approaches used to capture user sessions through the Web server logs are heuristic. The heuristic strategy only exploits the background knowledge on user navigational behaviour to assess whether requests registered by the Web server can belong to the same individual and whether these requests were performed during the same or subsequent visits of the individual to the site. However, the computations are affected by the use of one or more proxy servers, caching and stateless service model of the HTTP protocol [2,6]. None of the several existing heuristics addresses all of these problems. We propose a new heuristic to overcome these limitations.

D. Lowe and M. Gaedke (Eds.): ICWE 2005, LNCS 3579, pp. 63–68, 2005.

2 Related Works

Cooley et al [2,3] propose four heuristics for the attribution of requests to different users. We denote them as **h1**, **h2**, **h3** and **h4**. Heuristic **h1** states that *each different user-agent type for an IP address represents a different user.* A user-agent identifies the browser version and the operating system. The rationale here is that a user rarely employs more than one browser when navigating the Web. Hence, a user session is defined by aggregating accesses on unique user agents for an IP address. However, the algorithm ignores the possibility of the accesses representing more than one active session, for example through multiple browser windows, for a specific user over time. Heuristic **h2** states that *If a Web page is requested and this page is not reachable from previously visited pages, then the request should be attributed to a different user. A new session is also suspected if the referrer is undefined.* A referrer is the URL of the page the client was on before requesting the current page. The rationale behind this heuristic is that users generally follow links to reach a page. However, this overlooks the use of bookmarks or explicit typing of URL to reach pages not connected via links in which case the referrer is not available. So, this heuristic might misclassify these accesses as requests from different users. Heuristic **h3** states that *the duration of a session must not exceed a pre-specified threshold.* The threshold is an upper bound on the time spent in the site during a visit. Heuristic **h4** states that *a new session is suspected if the time spent on a page exceeds a pre-specified time threshold.* Users who do not request pages within a certain time limit are assumed to have left the site. Researchers have used some or all of the above heuristics to identify user sessions. In [8], IP and user agent are used to identify unique users and a time period of 6 hours is used as an upper bound for a session. In [1], IP addresses are used to identify users and their consecutive page requests are grouped by using both a session upper bound and a page upper bound.

3 Issues in Heuristic Based Approaches

The identification of user sessions directly from the server logs using heuristics is also affected by the use of proxy server(s), caching and statelessness of HTTP protocol.

The process of user session identification is usually performed mostly based on the IP address [1]. When a user's browser makes a request, the request is routed through a proxy at the ISP. The IP address that is then seen at the Web site is the IP address of the ISP's proxy, not the user's machine. Further complicating the matter, the ISP may have a number of proxies and user requests may go through different proxies. Thus, requests made by a single user (in a single visit) may have multiple IP addresses and one IP address may 'hide' multiple users.

Web browsers and proxy servers frequently cache the pages that have recently been accessed and meet the subsequent requests to these pages from the caches. There are no corresponding log entries for those accesses to the cached pages. Estimating their effect on capturing user sessions from the server logs is non-trivial.

The HTTP protocol is stateless as every request from the client to the server is treated independently and information from previous connections to the server is not maintained for use in future connection [5]. Thus, it does not allow support for establishing long-term connections between the Web server and the client. Therefore, the

requests of a single user are recorded in the server logs nested with the requests of other users. As such, clustering of server log entries into its sessions is also non- trivial.

4 A New Heuristic

A new heuristic proposed in this paper is a combination of four heuristics. The four heuristics are termed as H1, H2, H3 and h4. The first three heuristics are the results of our own analysis of Web server logs and introduced here. The last is the existing heuristic described in section 2.

H1: *Each different user agent for a domain of the IP addresses represents a different user sessions.*

In general, an IP address represents a domain and a server. As described above, an ISP may have one or more proxy servers but they will all belong to a single domain. A single IP address may appear in multiple sessions in which case different user agents identify separate sessions. The heuristic h1 described in section 2 is similar to this and the IP address itself may be used instead of the domain. When multiple IP addresses appear in a single session but belong to the same ISP, they are assumed to share the same domain and, again, different user agents identify distinct user sessions.

H2: *Let p and q be two consecutive page requests in a session S identified by heuristic H1. Let also r(q) and ip(q) be the referrer and IP address of q respectively. The membership of q in S is confirmed if one of the following three conditions is satisfied.*

1. *If r(q) is equal to r(p) or p, or r(q) was previously invoked within S*
2. *If r(q) is undefined and q was previously invoked within S*
3. *If r(q) is undefined and ip(q) does not represent a common proxy server*

Otherwise, q belongs to a new session.

Heuristic H1 ensures that pages requested by the same user are not separated into different sessions provided that s/he uses only one browser and one operating system. As the market for both browsers and operating systems gets ever more consolidated, it is highly likely that different users coming behind the same ISP will have the same user agent. H1 is unable to decipher this situation. Heuristic H2 further checks each session identified by H1 for the validity of the membership of its entries within itself.

Assuming that users follow hyperlinks to reach a page, each access pair of the referrer page and the requested page constitute a connected traversal path. That is, if none of the pages is brought from the cache, *r(q)* should be equal to *p*. Otherwise, we assume that the user must have accessed a cached page connecting *p* and *q*. These hits are missed in the server log. But these missing cache hits must have accessed the server in the recent past. Hence, heuristic H2 does not need the requested page to be accessible from the page immediately accessed before it to confirm the memberships.

When *r(q)* is undefined, the heuristic assures the membership of *q* in the session depending on whether *q* was previously invoked or *ip(q)* does not represent a common proxy server. Users use bookmarks or type URLs to request pages in which case the referrer is undefined. It can be argued that the request page of this type might have been previously visited in the recent history. A proxy server is said to be common if numerous overlapping requests from multiple users come behind it. If *ip(q)* does not

represent such a common proxy server, it is assumed that q should be part of *S* as each of the requests of *S* carries the same user agent and IP address.

H3: *All consecutive requests that are invoked within a small time interval belong to the same session.*

User requests for one URL frequently result in multiple entries in the server logs representing requests for the hyperlinked elements, such as images, style sheets and so on. As they are automatically downloaded due to the HTML tags, the time spans between them is very small and it is possible for a log to reflect an inaccurate ordering of them that the request for the referrer page follows the requested page. H2 will identify them into separate user sessions although the user agent and IP address remain the same. Hence, it is reasonable to group all the consecutive requests invoked within a small time interval as part of the same session. Two different time thresholds are used for common and non-common proxy servers as the numerous overlapping requests can come behind the common proxy servers in very short time intervals.

5 Experimental Environments and Set Up

The Web logs used in this investigation came from the server for a student lab used exclusively to teach two Internet-related subjects. The students have to create a Web site each and then learn scripting for both client-side and server-side processing. Each student is given an id and Web space. The server runs Microsoft Internet Information Server 5.0 on Windows 2000 advanced server platform. The lab has 20 workstations, each with a unique, hard-wired IP address. These machines primarily run Windows 2000 professional. Students access the server from the special lab, other labs or from outside, using either the university dial-up lines or some ISP. The university routes its traffic through two proxies. The university semester runs for 16 weeks during which time the students typically complete two assignments, some quizzes and a mini-project each. The lab has been running for almost seven years. We chose the first semester of 2003, viz. March-June 2003 that was the latest semester when we launched our experiments. Approximately 500 students enrolled in the two subjects. We examined the Web server logs of the entire semester. The server logs are created daily, starting at 10.00 AM and go on for the next 24 hours and the size of the log files ranged from over 1 MB to more than 77 MB.

Web server logs, in general, easily reach tens of megabytes per day, which causes the session identification process to be really slow and inefficient without an initial cleaning task. The cleaning process employed here performs the following tasks. First, the log entries referring to images are removed, based on the suffixes such as gif, jpg, jpeg, and png. Second, log entries with server response codes 4xx and 5xx are removed since they are client and server errors respectively. Third, robot accesses in the server logs are removed. Finally, the entries for the system administrator and tutors who mark the student assignments are eliminated, since we want to analyse only student sessions. The cleaned data is the base to which the heuristic is applied.

6 Results

The performance of the new heuristic is first evaluated by comparing the number of sessions produced by it to the ‘exact’ number of sessions, derived from a combination

of explicit user id and heuristic h4, defined above, to split the students' activities. This combination is assumed to be the best approximation for the exact sessions as user id uniquely identifies every user. The 'exact' method is called M1 and the method of constructing sessions using the new heuristic M2. Then, the new heuristic is compared with two combinations of existing heuristics, (h1, h4) and (h1, h2, h4), similarly. These are called methods M3 and M4 respectively. The time thresholds of 15 minutes and 3 and 16 seconds have been taken as the specified thresholds in h4 and the new heuristic respectively. Table 1 shows the statistics of sessions produced by each method and log entries (Hits). The percentages of differences between total sessions identified by M1=19694 and those of M2, M3 and M4 defined by Si=((M1-Mi)/M1)x100, where i=1,3 are recorded as S1=-11.68, S2=-14.72 and S3=-491.30. According to Table 1, the new heuristic overestimates the exact method with the minimum average and percentage difference than the other three methods.

Table 1. Number of Sessions Produced and Statistics

Item	Hits	M1	M2	M3	M4
Total	2,201,304.00	19,694.00	21,995.00	22,592.00	116,451.00
Average per day	20,195.45	180.68	201.79	207.27	1,068.36
Maximum per day	2,73,677.00	1,583.00	1,789.00	1,827.00	15,713.00
Minimum per day	420.00	5.00	5.00	5.00	15.00

7 Chi-Squared Tests

The chi-squared test is used here to investigate the significance of three methods M2, M3 and M4 shown on Table 1 to the exact method M1. We take the null hypothesis as H_0: Two samples are from same distribution and alternative hypothesis as H_1: Two samples are not from same distribution. χ^2 values calculated over 109 observations for each test method are recorded in Table 2. Of methods M2 and M3, which are not significantly different to the exact method, M2 gives the smallest χ^2 value (largest P-value) concluding that the new heuristic best approximates the exact method.

Table 2. Chi-squared Test Results

Control	Test method	d.f	Observed χ^2 values	P-value	Significance at 5%
M1	M2	108	68.154	0.999	Insignificant
M1	M3	108	106.076	0.534	Insignificant
M1	M4	108	4144.802	0.001	Significant

8 Conclusions and Further Work

This paper has proposed a new heuristic that can be used to capture user navigation paths in the Web by exploring Web server logs and reported on the investigation into the efficiency of it. The investigation has confirmed that the new heuristic outperforms the existing heuristics and best approximates the exact method. However, this analysis is based on the daily logs that start at 10 am, when a good number of students are already at work. When a student starts his/her session before 10 am and continues after 10 am, that session is split into two logs. The latter log is short of some initial

accesses of many users and the referrer heuristic H2 is adversely affected by this. M2 may exhibit lesser deviation than here if we merged all the log files together and ran the analysis again. Our future work includes the evaluating the heuristic as a user-oriented log analysis tool.

References

1. Berendit, B. and M. Spiliopoulou, *Analysis of Navigation Behaviour in Web Sites Integrating Multiple Information Systems*, VLDB Journal, **9**(1): p. 56-75, 2000,
2. Cooley, R., B. Mobasher, and J. Srivastava, *Data preparation for mining world wide web browsing patterns*, In Journal of Knowledge and Information Systems, **1**(1): p. 5-32, February, 1999,
3. Cooley, R., B. Mobasher, and J. Srivastava, *Grouping web page references into transactions for mining world wide web browsing patterns*, In Knowledge and Data Engineering workshop: p. 2-9, 1997,
4. Elo-Dean, S. and M. Videros, *Data mining the IBM Official 1996 Olympics Web Site*, Workshop on Research Issues in data Engineering, 1997,
5. Iyengar, A., *Dynamic Argument Embedding: Preserving State on the World Wide Web*, IEEE Internet Computing, **1**(2): p. 50-56, March-April, 1997,
6. Pitkow, J., *In Search of Reliable Usage Data on the WWW*, In Ths Sixth International World Wide Web Conference, 1997,
7. Yan, T.W., M. Jacobsen, H. Garcia-Molina, and U. Dayal, *From User Access Patterns to Dynamic Hypertext Linking*, In Fifth International World Wide Web Conference, 1996,
8. Yao, Y.Y., H.J. Hamilton, and X. Wang, *PagePrompter: An Intelligent Agent for Web Navigation Created Using Data Mining Techniques*, Technical Report CS-2000-08, Department of Computer Science, University of Regina, 2000,
9. Zaiane, O.R., M. Xin, and J. Han, *Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs*, In Advances in Digital Libraries: p. 19-29, 1998,