Popular Web Hot Spots Identification and Visualization

D. Avramouli[2], J. Garofalakis[1,2], D. J. Kavvadias[3], C. Makris[2], Y. Panagis[1,2] and E. Sakkopoulos[1,2]

[1]RA Computer Technology Institute
Internet and Multimedia Technologies RU 5
61 Riga Feraiou Str. 26110, Greece

[2]University of Patras
Computer Engineering & Informatics Dept
26500 Patras, Greece

[3] University of Patras
Mathematics Dept
26500 Patras, Greece

E-mail: {avramuli, makri, panagis, kavadias}@ceid.upatras.gr, {garofala, sakkopul}@cti.gr

ABSTRACT

This work aims a two-fold contribution: it presents a software to analyse logfiles and visualize popular web hot spots and, additionally, presents an algorithm to use this information in order to identify subsets of the website that display large access patterns. Such information is extremely valuable to the site maintainer, since it indicates points that may need content intervention or/and site graph restructuring. Experimental validation verified that the visualization tool, when coupled with algorithms that infer frequent traversal patterns, is both effective in indicating popular hot spots and efficient in doing so by using graph-based representations of popular traversals.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics
H.5.4 [**Information Systems**]: Hypertext

General Terms

Algorithms, Management, Design, Experimentation.

Keywords

Access visualization, Maximal Forward path, Usage mining

1. INTRODUCTION

With the immense growth of the Internet usage, websites are being developed in an uncontrollable, ad-hoc manner, a fact frequently reflected to unpredictable visit patterns. Thus, a critical task for a website maintainer is to use enumerable metrics in order to identify substructures of the site that are objectively popular.

Web Usage Mining has emerged as a method to assist such a task. The fundamental basis for all mining operations entails processing web server access logfiles. In its most simplified manner, usage mining entails registering absolute page visits or identifying popular paths of information inside a website, by the means of logfile analysing software solutions such as Webtrends (http://www.webtrends.com), and Analog (http://www.analog.cx). When the goal is to detect popular structural website elements, more elaborate techniques have been devised. Due to space limitations only representative work is presented.

Significant work on converting server logfiles to valuable sources of access patterns has been conducted by Cooley [4] and Chen et al. [1]. Cooley describes all the technical and algorithmic details to process and analyse logfiles for a variety of Web Mining tasks. Chen et al. describe efficient algorithms to infer access patterns corresponding to frequently traversed, website paths. Apart from analysing logfiles, it is important to use analysis as input and determine which changes, if any, to bring to the website structure. Srikant and Yang [5] infer path traversal patterns and use them to indicate structural changes that maximize (or minimize) certain site-dependent criteria. Finally, Christopoulou et al. [3] define metrics to assess the actual value of webpages and experiment on techniques to reorganize websites.

This work has two main merits: it presents a software to analyse logfiles and visualize "Maximal Forward Paths" (see [1]) and, additionally, it presents an algorithm to use this information in order to identify subsets of the website that display popular access patterns. Such information is valuable to the site maintainer, since it brings to attention web site "neighbourhoods" that may need content revisiting and structural redesign.

The rest of the paper is organized as follows. Section 2 introduces the popularity visualization tool. Section 3 describes the proposed algorithm to identify website parts that receive significant traffic. Section 4 concludes and presents future directions.

2. PAGE POPULARITY VISUALIZATION

This section describes functional characteristics and implementation aspects of the visualization software.

The developed software can perform structural and statistical analysis of a given website, using a GUI (see Figure 1). It supports a number of log input types such as W3C, IIS, NCSA, ODBC logs in typical or extended format. An HTML parser and a link crawler were implemented, hence, given the homepage of a website the link structure is discovered. Furthermore, the user can provide, in the corresponding window, the website logfile. The logfile is analysed to discover "Maximal Forward Paths" and "Forward Paths", as defined in [1]. The user can choose to display the site graph and the most frequent paths, each with a different colour. Frequent paths can be highlighted either on top of the Website layout or be presented standalone. In this way, possible recommendations can be identified quickly and directly on the site visual representation.

All processing class modules were implemented in the Java platform. Java 1.2 or higher is required to run the software. Both the site structure and the information resulting from logfile processing are stored as relational data in a DBMS. The choice of the DBMS is not restrictive; any database system can work as long as the communication is carried out with the means of a JDBC/ODBC driver. The development was based on the logfiles generated from an experimental web site running on Apache 1.3.27. However, the final version was verified using different logfile versions both on Windows & Linux servers.

WWW 2005, May 10--14, 2005, Chiba, Japan.
ACM 1-59593-051-5/05/0005.

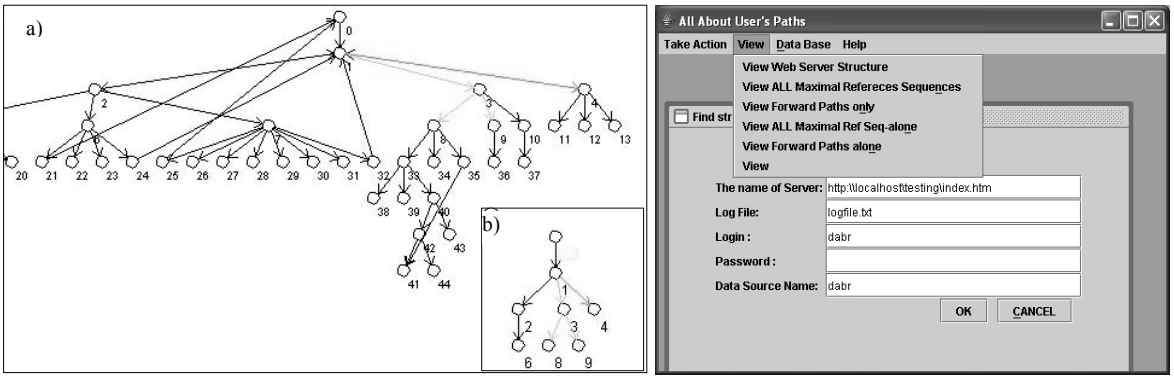

Figure 1 [left fig] (a) Forward paths on sample site. (b) Maximal Forward paths only, [right fig] Visualization UI

The task of visualization relies on the "Visualizing Graphs with Java" (VGJ) library. VGJ provides all the necessary functionalities to produce straight-line, planar embeddings of directed graphs. We have exploited the feature of VGJ to produce graph visualizations, given a graph description in GML format.

2.1 Algorithms and Experiments

For experimental verification, we have created a synthetic website, with no particular content, consisting of 45 nodes. Subsequently, site traversals were generated in the laboratory with human intervention and favouring specific site parts.

Since logfile processing is a critical step (data cleaning, session identification) the, widely accepted, conventions of ([4],[5]) were implemented. For identifying and visualizing interesting traversal patterns, we have implemented the algorithms in [1]. The authors in [1], describe efficient algorithms to combine information about user sessions and page hits, and produce frequently traversed paths. In the sequel, a novel algorithm that identifies popular hot spots has been devised thereby allowing multiple levels of recommendations to be coloured on the analysed site graphs.

3. IDENTIFICATION OF HOT SPOTS

In this section, we describe our proposed algorithm to identify parts of the website that receive significant traffic. Our algorithm scans the website graph in order to keep only the non-intersecting subpaths. Suppose that the site is modelled as a graph G(V,E) kept in an adjacency matrix representation, with matrix A. We postulate that nodes in V are numbered $1..n$.

First, we run the algorithm to identify "Maximum Forward Paths". The algorithm of [1], keeps website access sequences (paths) where each path sequence P_i, of access frequency count(P_i), occurs with count(P_i)≥min{support}. In order to keep only these paths we set to zero the corresponding cells in A, thereby pruning the remaining paths. Suppose that there are p frequent paths. We keep an array C[$1..p$], such that C[i] = count(P_i).

A breadth-first search is initialised. Suppose we visit node v. If there is only one outgoing edge from v in the graph remaining after the initial pruning, then there must be a single non-zero element in the v-th row of A. If this is the case, we delete node v by zeroing out the non-zero entry. There is a small subtlety here; v might be part of a larger path. Therefore, we delete v only if there is no entry but zeros in the v-th column of A (an indication that no edge ends at v). When deleting node v we add the path frequencies of the paths passing through v, to their subpaths. For example suppose that we had a path $P_i = abfh$ and paths $P_j = bfh$, $P_k = bh$. After elimination of a we perform C[j] = C[j]+C[i] and C[k] = C[k]+C[i]. The procedure stops when there are no nodes left to be eliminated. The remaining non-zero elements correspond to a subgraph of G with high popularity.

The above algorithm can gracefully adjust to multiple levels of granularity. After setting the threshold at the first path-elimination step, then after each vertex-elimination, one can define a higher threshold to correspond to more important paths, and so on. In the visualization tool the three most popular paths are coloured (observe the lightly shaded paths at left fig of Figure 1).

4. CONCLUSIONS

We have presented a software that can process website logfiles and visualize site structure and the access patterns imposed on it. In the sequel we devised an algorithm to isolate parts of the site subgraph that contain important access information. We believe that identifying those parts can help website maintainers to perform local structural changes. Automation of such structural changes can be achieved using the authors' proposed concepts in [2]. Nevertheless, the very nature of changes and their effects remain issues that need to be further explored. As a future step, we intend to integrate the visualization tool with web modelling and CASE tools that would facilitate site reorganization algorithms in a semi-automatic or full-automatic way.

5. REFERENCES

[1] M.-S. Chen, J. S. Park, and P. S. Yu. Efficient Data mining for path traversal patterns. *Knowledge and Data Eng.*, 10(2), pp. 209–221, 1998.

[2] E. Christopoulou, J.Garofalakis, C. Makris, Y. Panagis, E. Sakkopoulos and A.Tsakalidis, Automating Restructuring of Web Applications, in *13th ACM Hypertext, Poster,* 2002*,* available at: *http://mmlab.ceid.upatras.gr/ht02/ht2002.pdf.*

[3] E. Christopoulou, J.Garofalakis, C.Makris, Y. Panagis, E. Sakkopoulos, A. Psaras-Chatzigeorgiou and A.Tsakalidis, Techniques and Metrics for Website Reorganization, *J. of Web Eng.*, 2(1-2), pp. 90-114, 2003.

[4] R. Cooley. *Web Usage Mining: Discovery and Application of Interesting Patterns from Web data*. PhD thesis, University of Minnesota, 2000.

[5] R. Srikant, Y. Yang, Mining Web Logs to Improve Web Site Organization, in *Proc. WWW01*, pp. 430-437, 2001.