Open access • Proceedings Article • DOI:10.1109/MASCOT.2002.1167096

WebTraff: a GUI for Web proxy cache workload modeling and analysis

— Source link

Nayden Markatchev, Carey Williamson

Institutions: University of Calgary

Published on: 11 Oct 2002 - Modeling, Analysis, and Simulation On Computer and Telecommunication Systems

Topics: Proxy pattern, Cache, Web page, Locality of reference and Graphical user interface

WebTraff: A GUI for Web Proxy Cache Workload Modeling and Analysis

Nayden Markatchev   Carey Williamson
Department of Computer Science
University of Calgary
E-mail: {nayden,carey}@cpsc.ucalgary.ca

Abstract

This paper describes an interactive graphical user interface (GUI) that can be used for the modeling and analysis of Web proxy workloads. The WebTraff GUI has three main components. First, the WebTraff tool provides a visual front-end to ProWGen, a Web proxy workload generation tool developed in prior work, which can be used for generating synthetic Web proxy workloads of arbitrary length, with user-specified statistical properties. Second, the WebTraff GUI provides tools for the analysis of Web proxy workload characteristics, including document size distribution, document popularity profile, and temporal locality properties. Third, the GUI provides a front-end to a simple Web proxy cache simulation program, which can be used in studies of Web proxy cache performance and cache filter effects.

1. Introduction

Traffic modeling is an essential part of any network simulation study, such as a performance study of Web proxy caching architectures. Controllable and representative workloads are crucial for these studies, for successful realization of cost-effective designs and for a thorough evaluation of their performance sensitivities.

In previous work [9], we developed a tool called ProWGen (Proxy Workload Generator) for the synthetic generation of Web proxy traces, with controllable workload characteristics. We also used this tool in a simulation study evaluating the sensitivity of Web proxy cache performance to certain workload characteristics [10].

The purpose of this paper is to describe a visually interactive front-end for traffic modeling in Web caching performance studies. This toolkit unifies a disparate set of tools used previously for modeling and analysis of Web proxy workloads, and for Web proxy cache simulation. By harnessing these tools together with a unified graphical user interface (GUI), we improve the usability of the tools. We also make the tool set available to a broader set of performance modelers for use in Web proxy caching studies.

The remainder of this paper is organized as follows. Section 2 provides an overview of the WebTraff toolkit (e.g., system requirements, trace format, inputs, and outputs). Section 3 describes the three main parts of the *WebTraff* toolkit for generation, analysis, and simulation of Web proxy cache workloads. Finally, Section 4 summarizes the paper and the current status of our toolkit.

2. General Background Information

2.1. Overview of WebTraff

The WebTraff toolkit provides three main functions: Web workload trace generation, Web workload trace analysis, and Web proxy cache simulation. These functionalities can be used either collectively or in isolation. For example, if an empirical Web workload trace is available from a Web proxy access log, then the workload generation phase can be bypassed, with the (appropriately pre-processed) empirical workload trace used directly in the analysis and/or simulation phases. Conversely, if no empirical data sets are available, the generation phase can be used to make synthetic workloads of arbitrary length, with desired workload characteristics. The analysis functions can be used to verify the properties of the workload trace prior to running Web cache simulation experiments. Finally, the output trace of misses from the Web proxy cache simulation step can be used (recursively) in the analysis and simulation phases, to study cache filter effects in Web caching hierarchies [4, 5, 15]. Graphical results produced from the analysis and/or simulation stages can be saved to a file in PostScript form, and used when writing research papers, such as this one.

2.2. Workload Trace Format

The WebTraff toolkit uses a simple three-column format for representing Web proxy workloads, as shown in Figure 1. Each line of data in the file represents a Web document (object) transfer. The first column is a timestamp, representing the time in seconds at which a specific Web object (i.e., URL) is requested. The second column is a document identifier: a unique integer assigned to each URL represented in the workload trace. The third column represents the size in bytes of the specific Web object. This size is fixed throughout the workload trace for a given document id, since only static Web content is modeled.

This trace format provides a simple and concise representation of Web proxy workloads. The file contains all information essential to a Web caching performance study (i.e., document id, transfer size, relative ordering of requests), while excluding other information (e.g., client id, URL name, document type). All generated traces use this format, which can also be obtained by extracting the appropriate columns of information from the Common Log Format used by most Web servers and Web proxies. The same trace format is used both for input and output traces. That is, the stream of misses from a cache simulation are recorded using the exact same trace format.

TIMESTAMP	DOC_ID	SIZE
0.03245	0	1958
2.73954	9	366
3.47710	4	2536
3.57192	0	1958
4.61692	26	82906
4.68677	3	3413
5.11075	21	3257
8.97064	4	2536
9.63967	5	2914
11.23907	23	149
11.30097	2	38735
12.89020	0	1958
13.86087	9	366

Figure 1. Example of the Web Workload Trace Format Used in WebTraff

2.3. System Requirements

Our tool has been developed in and for a Unix-based environment running X windows. The traffic generation, analysis, and simulation tools are written in C/C++ and perl. The user interface is written in Tcl/Tk.

Installing and running our tool requires the following software: cc, gcc, g++, tcl (version 8.0 or newer), tk (version 8.0 or newer), wish, perl (version 5.0 or newer), gnuplot (a graph plotting tool), and gs (a PostScript previewer, also known as ghostscript, gsview, or gsview32).

For hardware requirements, at least 64 MB RAM is desirable, so that reasonably large data sets can be analyzed (e.g., 1-10 million references). Adequate disk storage capacity (e.g., 10-100 Megabytes) is required for storing workload traces, which tend to be large.

3. The WebTraff Tool

Figure 2 shows a screen shot of the WebTraff GUI. The interface has three main blocks corresponding to workload generation (top), workload analysis (middle), and Web proxy cache simulation (bottom). The following sections provide further details on the functionality provided in each of these three blocks.

3.1. Web Workload Generation

The top block of the WebTraff GUI is for generation of synthetic Web proxy workload traces. This portion of the tool simply provides a graphical interface to ProWGen [9], a Web proxy workload generation tool developed in prior work. ProWGen models the aggregate request stream that a Web proxy cache might see from many concurrent Web clients.

The ProWGen tool models four common characteristics that have been identified in empirical studies of Web server and Web proxy workloads [1, 2, 7, 11]. These characteristics include a *Zipf-like document popularity distribution* [8], a high degree of *one-time referencing* [2], *heavy-tailed file and transfer size distributions* [11], and a *temporal locality property* [1] in the document referencing behaviour. These characteristics are all relevant to Web proxy cache performance [10].

The top portion of the GUI allows the user to set parameters to control the workload characteristics, prior to hitting the 'Generate' button. An entry box at the top of the GUI specifies the name of the trace file to be generated. Sliding scale widgets are used to specify the number of references (requests) desired in the generated workload, as well as the number of distinct Web objects (expressed as a percentage of the total references), and the number of one-timers (expressed as a percentage of the total documents). Separate sliders are used to specify the slope for the Zipf-like document popularity distribution, the slope for the Pareto tail of the document size distribution (the body of the document size distribution is log-normal, with a median of 4 KB and a mean of 10 KB), and the degree of statistical correlation (if any) between the size and popularity of Web objects. Positive correlation means that larger objects are more likely to be referenced. Negative correlation means that smaller objects are more likely to be referenced. The default setting of zero correlation means that document size and document popularity are independent characteristics.

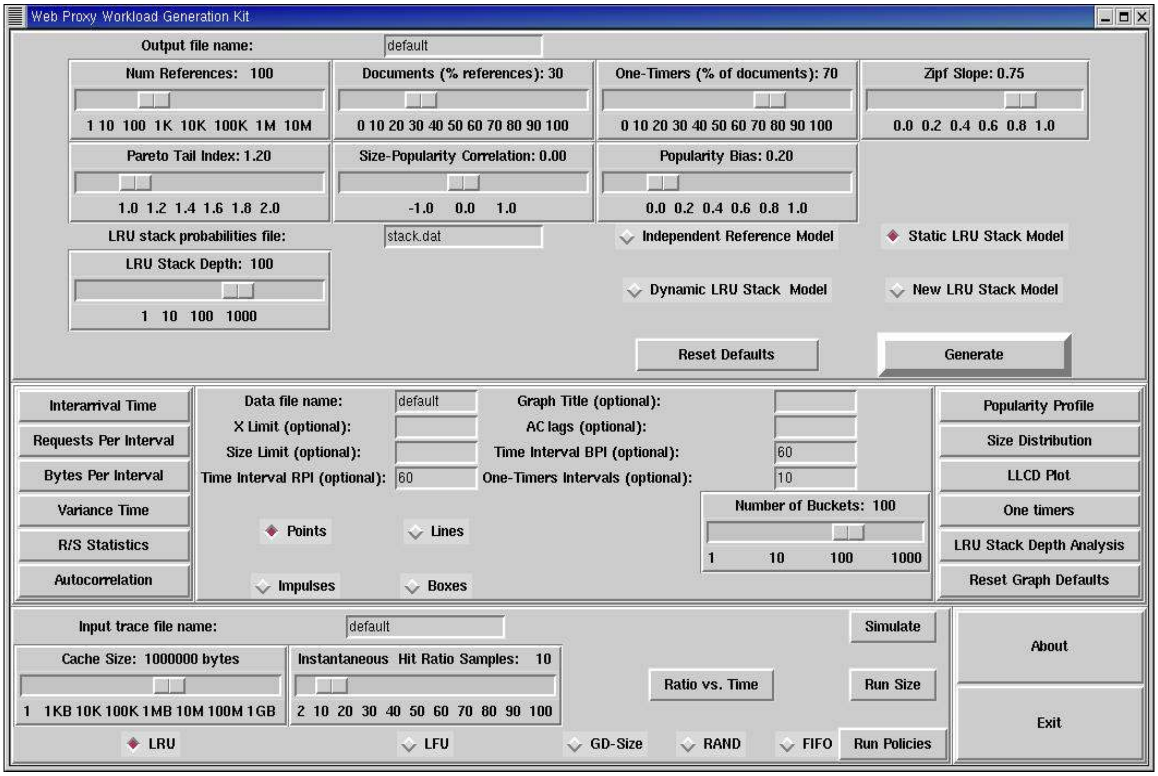

Figure 2. Screen Shot of Graphical User Interface (GUI) for WebTraff Tool

One novel aspect of the WebTraff GUI is a choice of temporal locality models to use when generating a synthetic workload. Temporal locality refers to the property that the recent past is often a good predictor of the near future, in terms of which Web objects are referenced next. The temporal locality model affects the relative order in which references appear in the generated workload. Clearly, this ordering can have a large impact on Web proxy cache performance [1, 10, 13].

There are three temporal locality models available in WebTraff. The first is the Independent Reference Model (IRM). This simple model assumes that the inter-reference times for a particular Web object are drawn independently from a geometric distribution. There is no notion of hysteresis (past history) in this model; references are generated independently to each document based on relative document referencing probabilities. The remaining temporal locality models are all based on the Least-Recently-Used (LRU) stack model (LRUSM). In the LRUSM, a stack data structure is used to store the recent history of the reference stream generation. In particular, the stack stores the N most recently referenced items (where N is specified by the user), with the most recent item at the top of the stack. Hysteresis is achieved by associating a probability with referencing a document on the stack (as opposed to the general pool of available documents) when generating the next request. The user must provide the name of an external file that contains the desired stack reference probabilities, which are typically determined by analyzing an empirical trace.

The temporal locality models differ in how the stack probabilities are computed and used. In the static LRU stack model, the reference probabilities are (statically) associated with particular stack levels, regardless of which document currently resides at that position (if any). In the dynamic LRU stack model, the reference probabilities are associated with particular documents, and thus the stack level reference probabilities change (dynamically) with time depending on which documents are stacked. These models are effective for modeling the temporal locality characteristics in empirical Web proxy workloads [9, 13].

One final control parameter in the WebTraff GUI is the ‘Popularity Bias’ button, which can influence whether popular documents tend to get chosen early or late in the synthetic trace generation. This button was added to remedy a problem in an early version of ProWGen, wherein the temporal locality model tended to choose most of the one-timer documents early in the trace and most of the popular documents late in the trace. This behaviour resulted in non-stationary cache hit ratio performance throughout the trace. This new button allows the user to control this behaviour.

The default setting of the Popularity Bias is adequate for most traces, but the slider may need to be adjusted if unusually long traces are being generated.

Each click of the 'Generate' button produces an independent sample path realization of the specified Web proxy workload characteristics. Currently, the request arrival process is modeled as Poisson [2, 14], though other models can be added.

3.2. Web Workload Analysis

The middle block of the WebTraff GUI is for workload analysis. The analysis functions fall into two main categories: time series analysis (on the leftmost edge of the middle block), and Web workload analysis (on the rightmost edge of the middle block). The user must specify the name of the workload file to be analyzed. The default file name is the same as the most recently generated workload file.

The time series analysis tools work primarily with the first column of the Web workload trace, namely the timestamp (see Figure 1). The buttons here produce graphs of the request inter-arrival time distribution, the request count per (user-specified) time interval, and the byte count per (user-specified) time interval. Examples of the graphs produced by these buttons are shown in Figure 3. Additional buttons provide tests for long-range dependence (LRD) in the request arrival process (i.e., autocorrelation function, variance-time plot, and R/S analysis) [12, 14]. These analyses are similar to those supported in the *SynTraff* GUI developed previously for modeling and analysis of LRD traffic [6].

The Web workload analysis buttons work primarily with the second and third columns of data, namely the document id and size. The buttons here analyze the document size distribution, the transfer size distribution, and the tail of the size distributions (using a log-log complementary distribution (LLCD) plot). The latter analysis is useful for testing for a heavy-tail in the size distribution [11]. Two additional buttons produce analyses of the document popularity profile (i.e., to check for a Zipf-like popularity profile) and the LRU stack depth referencing behaviour (i.e., to characterize temporal locality properties).

Most of the analysis buttons generate one or more graphs in a "pop up" fashion on the screen for the user. The user has radio buttons to control graph plotting characteristics (e.g., points, lines, boxes, impulses) as well as the graph caption and the numerical ranges for axes. Additional slider buttons control the granularity of certain analyses (e.g., number of buckets in marginal distribution plots). Examples of the graphs produced by the Web workload analysis buttons are shown in Figure 4.

3.3. Web Proxy Cache Simulation

The bottom block of the WebTraff GUI is for simulation studies of Web proxy cache performance. The Web cache simulator used is a simple application-level document caching model. Two parameters are required: the size of the cache, and the cache replacement policy to be used to remove documents when more space is required to store an incoming document (e.g., Random replacement, First-In-First-Out, Least-Recently-Used, Least-Frequently-Used, and Greedy-Dual-Size) [3]. The user specifies these two parameters, as well as the name of the Web proxy workload file to be used as input for the trace-driven simulation.

The remaining buttons invoke different types of simulation experiments. The 'Simulate' button runs a single simulation with the user-specified cache size and replacement policy. As a side effect, it produces a file with the stream of cache misses from the simulation. The 'Run Size' button runs a set of simulations for a given cache replacement policy, with a range of cache sizes specified by the user. The resulting graph plots the document hit ratio (HR) and the byte hit ratio (BHR) as a function of cache size. The 'Run Policies' button runs a set of simulations that varies the cache replacement policy as well as the cache size. Two graphs are created for comparing replacement policy performance: one for the HR and the other for the BHR. Finally, the 'Hit Ratio vs. Time' button provides a visual look at the stationarity (or non-stationarity) of the hit ratio and byte hit ratio as a function of simulation time within the trace. This plot is useful for identifying warmup and end effects, and any anomalies present in the input workload. This button uses the LRU replacement policy only, and a range of cache sizes up to the maximum cache size specified by the user.

Examples of the graphs produced by the simulation part of the WebTraff GUI are shown in Figure 5.

4. Summary

This paper has presented a graphical user interface for the generation and analysis of Web proxy workloads, and for simple simulation studies of Web proxy cache performance. The Web traffic modeling and analysis tools are based on our prior work in the published literature [6, 9], and have an easy-to-use graphical interface written in Tcl/Tk. We have found the toolkit useful in a sensitivity study of Web proxy cache performance [10], and in a study of filter effects in Web caching hierarchies [4, 5, 15].

The WebTraff toolkit is available to the MASCOTS community for research purposes. The URL is:
`www.cpsc.ucalgary.ca/~carey/software.htm`
At this time, only a UNIX-based version of the tool is available. We would welcome an effort to port this tool to the PC Windows environment, to enable wider usage.

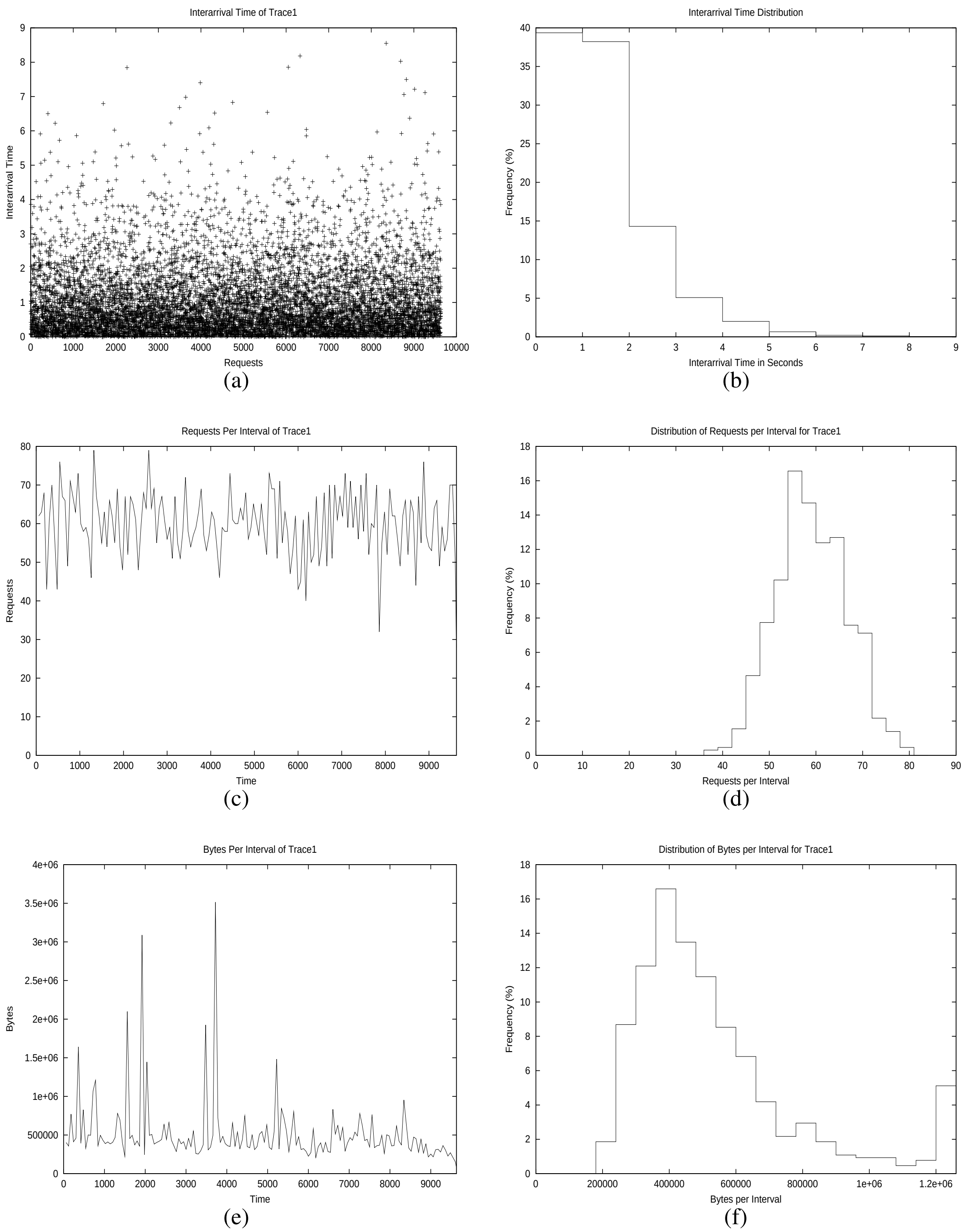

Figure 3. Example of Time Series Analyses in WebTraff: (a) 'Inter-Arrival Time' Time Series; (b) 'Inter-Arrival Time' Distribution; (c) 'Requests per Interval' Time Series; (d) 'Requests per Interval' Distribution; (e) 'Bytes per Interval' Time Series; (f) 'Bytes per Interval' Distribution

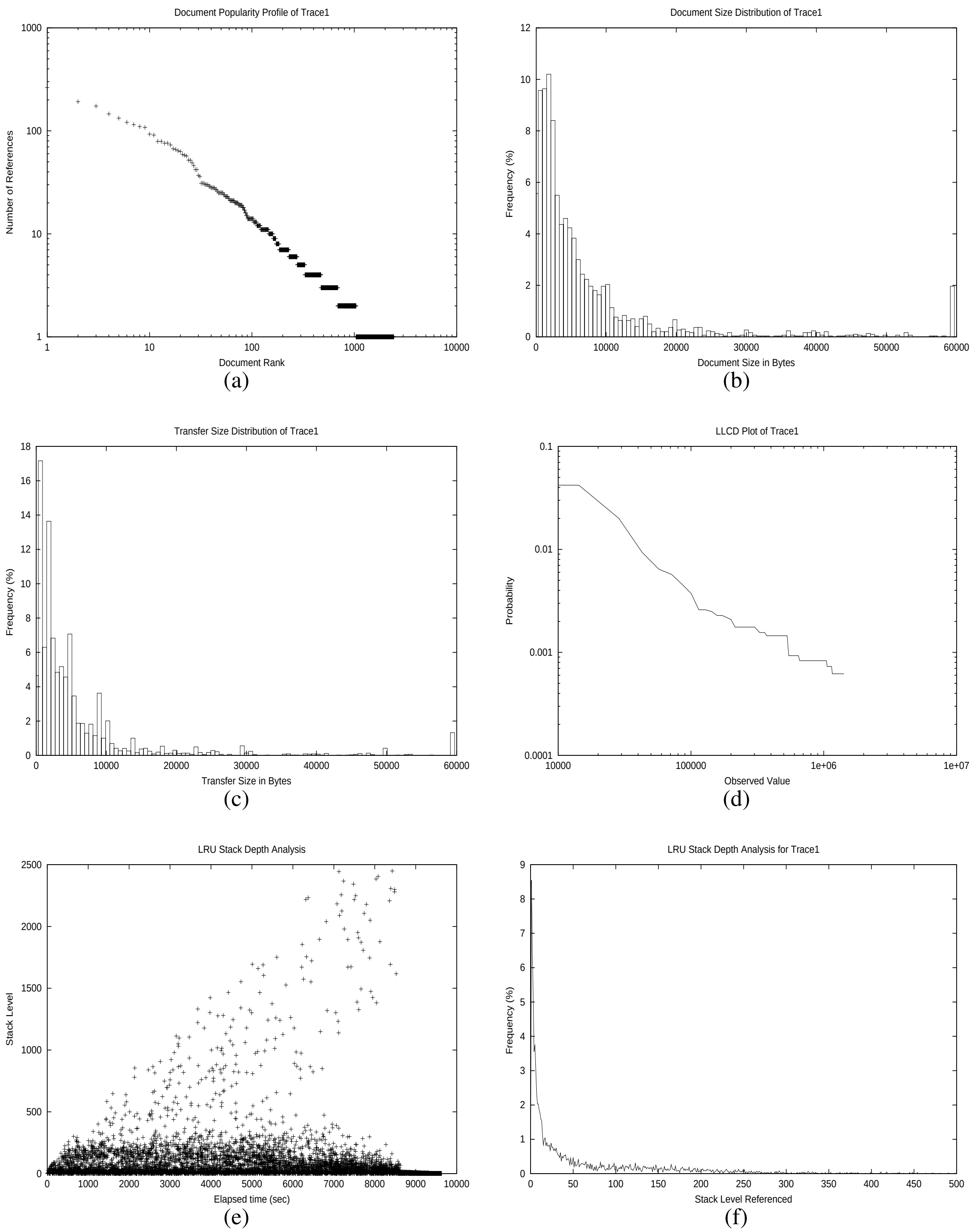

Figure 4. Example of Web Workload Analyses in WebTraff: (a) Zipf-Like Document Popularity Profile; (b) Document Size Distribution; (c) Transfer Size Distribution; (d) LLCD Plot of Transfer Size Distribution; (e) LRU Stack Depth Time Series; (f) LRU Stack Depth Distribution

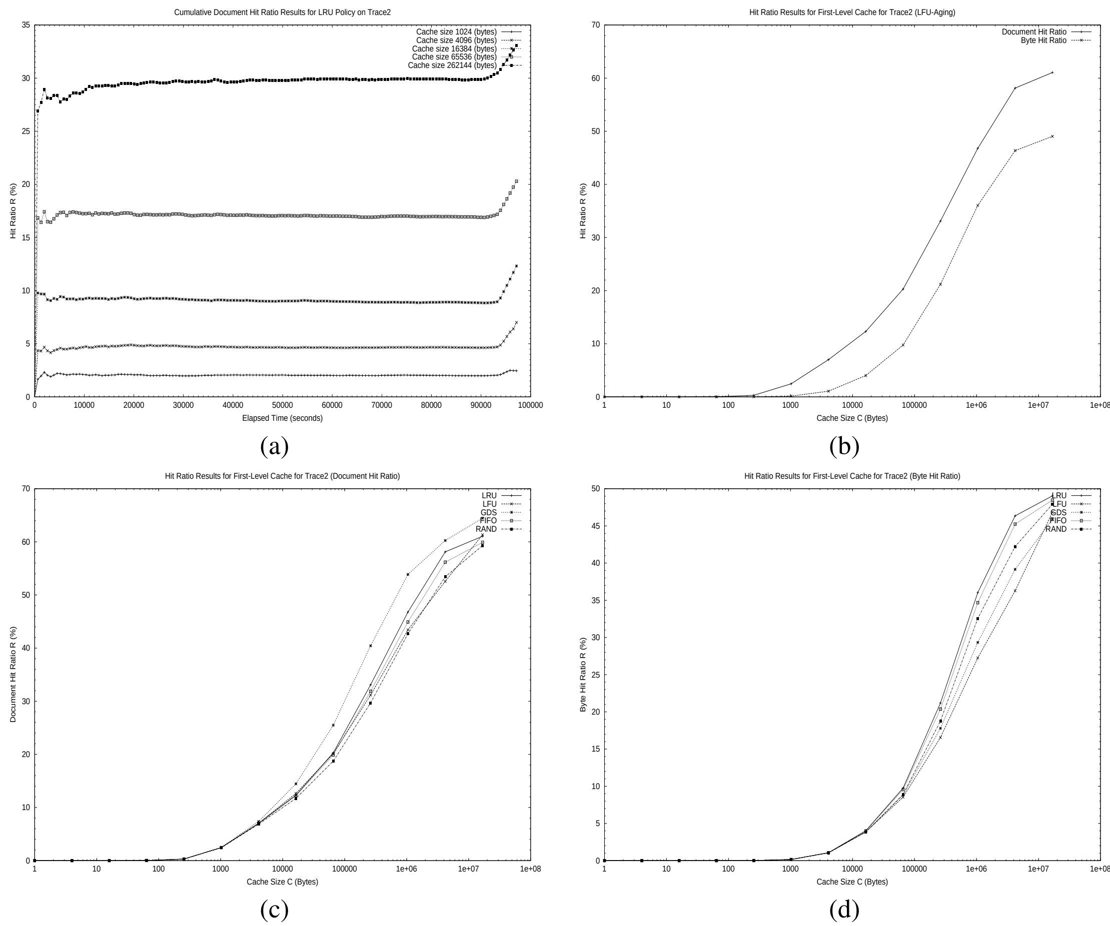

Figure 5. Example of Web Proxy Cache Simulation Results in WebTraff: (a) 'Hit Ratio versus Time' Plot for LRU Policy; (b) 'Run Size' Results for HR and BHR versus Cache Size; (c) 'Run Policies' Results for HR versus Cache Size; (d) 'Run Policies' Results for BHR versus Cache Size

Acknowledgements

Financial support for this research was provided by iCORE (Informatics Circle of Research Excellence) in the Province of Alberta, and by the Natural Sciences and Engineering Research Council of Canada, through NSERC Research Grant OGP0121969.

References

[1] V. Almeida, A. Bestavros, M. Crovella, and A. Oliveira, “Characterizing Reference Locality in the WWW”, *Proceedings of the 1996 International Conference on Parallel and Distributed Information Systems (PDIS’96)*, pp. 92-103, December 1996.

[2] M. Arlitt and C. Williamson, “Internet Web Servers: Workload Characterization and Performance Implications”, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 5, pp. 631-645, October 1997.

[3] M. Arlitt and C. Williamson, “Trace-Driven Simulation of Document Caching Strategies for Internet Web Servers”, *Simulation Journal*, Vol. 68, No. 1, pp. 23-33, January 1997.

[4] G. Bai and C. Williamson, “Time-Domain Analysis of Web Cache Filter Effects”, *Proceedings of SCS International Symposium on Performance Evaluation of Computer and Telecommunication Systems* (SPECTS’02), San Diego, CA, pp. 195-205, July 2002.

[5] G. Bai and C. Williamson, “Workload Characterization in Web Caching Hierarchies”, *Proceedings of IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems* (MASCOTS), Fort Worth, TX, October 2002.

[6] R. Balakrishnan and C. Williamson, “The *synTraff* Suite of Traffic Modeling Toolkits”, *Proceedings of IEEE MASCOTS*, San Francisco, CA, pp. 333-340, August 2000.

[7] P. Barford and M. Crovella, “Generating Representative Web Workloads for Network and Server Performance Evaluation”, *Proceedings of ACM SIGMETRICS Conference*, Madison, WI, pp. 151-160, June 1998.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web Caching and Zipf-like Distributions: Evidence and Implications”, *Proceedings of the IEEE INFOCOMM Conference*, New York, NY, pp. 126-134, March 1999.

[9] M. Busari and C. Williamson, “ProWGen: A Synthetic Workload Generation Tool for Simulation Evaluation of Web Proxy Caches”, *Computer Networks*, Vol. 38, No. 6, pp. 779-794, June 2002.

[10] M. Busari and C. Williamson, “On the Sensitivity of Web Proxy Cache Performance to Workload Characteristics”, *Proceedings of IEEE INFOCOMM*, Anchorage, AL, pp. 1225-1234, April 2001.

[11] M. Crovella and A. Bestavros, “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes”, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 835-846, December 1997.

[12] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, “On the Self-Similar Nature of Ethernet Traffic (Extended Version)”, *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, pp. 1-15, February 1994.

[13] A. Mahanti, D. Eager, and C. Williamson, “Temporal Locality and its Impact on Web Proxy Cache Performance”, *Performance Evaluation*, Vol. 42, No. 2-3, pp. 187-203, October 2000.

[14] V. Paxson and S. Floyd, “Wide Area Traffic: The Failure of Poisson Modeling”, *Proceedings of the 1994 ACM SIGCOMM Conference*, London, UK, pp. 257-268, August 1994.

[15] C. Williamson, “On Filter Effects in Web Caching Hierarchies”, *ACM Transactions on Internet Technology*, Vol. 2, No. 1, pp. 47-77, February 2002.