Web Server Workload Characterization: The Search for Invariants (Extended Version)[*]

Martin F. Arlitt Carey L. Williamson
Department of Computer Science
University of Saskatchewan
57 Campus Drive
Saskatoon, SK, CANADA S7N 5A9
{mfa126,carey}@cs.usask.ca
March 15, 1996

Abstract

The phenomenal growth in popularity of the World Wide Web (WWW, or the Web) has made WWW traffic the largest contributor to packet and byte traffic on the NSFNET backbone. This growth has triggered recent research aimed at reducing the volume of network traffic produced by Web clients and servers, by using caching, and reducing the latency for WWW users, by using improved protocols for Web interaction.

Fundamental to the goal of improving WWW performance is an understanding of WWW workloads. This paper presents a workload characterization study for Internet Web servers. Six different data sets are used in this study: three from academic environments, two from scientific research organizations, and one from a commercial Internet provider. These data sets represent three different orders of magnitude in server activity, and two different orders of magnitude in time duration, ranging from one week of activity to one year of activity.

Throughout the study, emphasis is placed on finding workload *invariants*: observations that apply across all the data sets studied. Ten invariants are identified. These invariants are deemed important since they potentially represent universal truths for all Internet Web servers. The paper concludes with a discussion of caching and performance issues, using the invariants to suggest performance enhancements that seem promising for Internet Web servers.

1 Introduction

The popularity of the World Wide Web [1, 22] (also called WWW, or the Web) has made Web traffic the fastest growing component of packet and byte traffic on the NSFNET network backbone [14]. WWW traffic has increased from 74 Megabytes per month in December 1992 to 3.2 Terabytes per month in December 1994.

There are many reasons behind this explosive growth in Web traffic. These reasons include: the ease of use of the Web; the availability of graphical user interfaces for navigating the Web; the availability of editors and support tools for creating and "publishing" Web documents; an emerging trend among researchers, educational institutions, and commercial organizations to make the Web the standard mechanism for disseminating information in a timely fashion; the machine-independent nature of the languages and protocols used for constructing and exchanging Web documents; and a continuing exponential increase in the number of Internet hosts and users [18].

The phenomenal and alarming growth in Web traffic has sparked much research activity on "improving" the World Wide Web. For example, researchers have proposed caching strategies for Web clients [3], caching strategies for Web servers [4], regional file caching strategies for large internetworks [8], and improved protocols for Web interaction [15, 20].

Much of this recent research activity has been aimed at improving Web performance and scalability. The key performance factors to consider are how to reduce the volume of network traffic produced by Web clients and servers, and how to improve the response time for WWW users.

Fundamental to the goal of improving Web performance is a solid understanding of WWW workloads. While there are several studies reported in the literature [3, 4, 6, 7, 12], most studies present data from only one measurement site, making it difficult to generalize results to other sites. Furthermore, most studies focus on characterizing Web clients, rather than Web servers.

The purpose of this paper is to present a detailed workload characterization study of Internet Web servers, similar to earlier studies of wide-area network TCP/IP

[*]A shorter version of this paper is to appear at the 1996 ACM SIGMETRICS Conference, Philadelphia, PA, May 1996.

Table 1: Summary of Invariants Found in Web Server Workloads

Invariant	Name	Description
1	Success Rate	Success rate for lookups at server $\approx$ 88%
2	File Types	HTML and image files account for 90-100% of requests
3	Mean Transfer Size	Mean transfer size $\leq$ 21 kilobytes
4	Distinct Requests	Among all server requests, less than 3% of the requests are for separate (distinct) files
5	One Time Referencing	Approximately one-third of the files and bytes accessed in the log are accessed only once in the log
6	Size Distribution	File size distribution is Pareto with $0.40 < \alpha < 0.63$
7	Concentration of References	10% of the files accessed account for 90% of server requests and 90% of the bytes transferred
8	Inter-Reference Times	File inter-reference times are exponentially distributed and independent
9	Remote Requests	Remote sites account for $\geq$ 70% of the accesses to the server, and $\geq$ 60% of the bytes transferred
10	Wide Area Usage	Web servers are accessed by 1000's of domains, with 10% of the domains accounting for $\geq$ 75% of usage

traffic [5]. Six different Web server access logs are used in this study: three from academic environments, two from scientific research institutions, and one from a commercial Internet provider. The data sets represent three different orders of magnitude in server activity, ranging from 776 requests per day to 355,787 requests per day, and time durations ranging from one week of activity to one year of activity.

Throughout the study, emphasis is placed on finding workload *invariants*: observations that apply or seem to apply across all the data sets studied. These invariants are deemed important since they potentially represent universal truths for all Internet Web servers.

Our research to date has identified ten invariants for Web server workloads. These invariants are summarized in Table 1, for easy reference, and are described in more detail within the paper itself.

The remainder of this paper is organized as follows. Section 2 provides background material on the World Wide Web, Web clients, and Web servers. Section 3 describes the Web server logs used in this study, and presents summary statistics for the six data sets. Section 4 presents the detailed results of our workload characterization, identifying the main invariants. The paper concludes, in Section 5, with a discussion of caching and performance issues for Internet Web servers, drawing upon the invariants to identify the types of performance enhancements that seem promising for Internet Web servers.

2 The World Wide Web

2.1 Web Overview

The World Wide Web is based on the client-server model. A client accesses documents on the Web via a Web browser. The browser sends a request to a Web server, which responds with the requested documents. Although the information may be stored in almost any location throughout the world, the Web provides the user with the illusion that the data is stored locally.

A Web server can respond to requests from multiple Web clients. Communication is always in the form of request-response pairs, and is always initiated by the client. Web clients and servers communicate using the HyperText Transfer Protocol (HTTP). HTTP runs on top of TCP, a reliable bidirectional byte stream protocol at the transport layer [21].

Communication between a Web client and a Web server is carried out in the following manner. When a client has a request to make of a particular Web server, the client must contact that server. A TCP connection must be established between the client and server, over which the request and response can be exchanged. Once the connection has been established, the client sends its request to the server. The server parses the request, and issues a response. Once the response is complete, the TCP connection between the Web client and server is closed. This process is repeated each time a client wishes to retrieve a document from a Web server [15].

2.2 Web Clients

A human user can gain access to the information on the World Wide Web by using a Web browser, such as netscape, mosaic, or lynx [22]. When the user selects a document to retrieve (usually by clicking a mouse on a hyperlink), the browser creates a request to be sent to the corresponding Web server. The request includes: the name of the requested document, expressed as a Uniform Resource Locator (URL) [2]; a set of HyperText request headers, indicating which data formats the client will accept; and user authentication information, which tells the server which documents the client has permission to retrieve. Once the request has been sent to the Web server, the client machine waits for a response. When the response arrives, the browser parses the reply. Depending on the response, the client machine may make another request to the server, or display the document for the human user to view.

2.3 Web Servers

The purpose of a Web server is to provide documents to Web clients that request them. Each Web *page* may consist of multiple documents (files). Each file is requested separately from the Web server.

A Web server operates as follows. The server listens on a designated port (usually port 80) for a request from a Web client to establish a TCP connection. Once a TCP connection has been opened and the client has made its request, the server must respond to that request. The response includes a status code to inform the client if the request succeeded. If the request was successful, then the response includes the requested document. If the request was unsuccessful, a reason for the failure is returned to the client [15]. Once the Web server has sent its response and terminated the TCP connection with the client, the server repeats the cycle and begins listening for its next request.

2.4 Server Logs

Web servers can be configured to record information about all client requests. Four log files are common to NCSA httpd version 1.4: an *access* log, an *agent* log, an *error* log and a *referer* log. The access log records information about all the requests and responses processed by the server. The agent log records the type of browser that was used by the client to issue the request. The error log records unusual Web server events that might require the attention of a Web master or system administrator. The referer log contains information on which Web pages (local or remote) are linked to documents on the (local) Web server. Only the access logs are used in the workload study reported in this paper.[1]

Each line from the access log contains information on a single request for a document. The log entry for a normal request is of the form:

```
hostname - - [dd/mmm/yyyy:hh:mm:ss tz] request status
bytes
```

From each log entry, it is possible to determine the name of the host machine making the request, the time that the request was made, and the name of the requested document. The entry also provides information about the server's response to this request, such as if the server was able to satisfy the request (if not, a reason why the response was unsuccessful is given) and the number of bytes transmitted by the server, if any.

[1]We used the error log from the University of Saskatchewan's Web server to study the aborted connections in the access log. Aborted connections are discussed in more detail in Section 4.4.

An example of a line from an access log is:

```
alfonso.usask.ca - - [15/Aug/1995:13:50:05 -0600]
        "GET / HTTP/1.0" 200 1265
```

This request came from the host `alfonso.usask.ca` at 1:50:05 pm CST on August 15, 1995. The requested document was the home page ("/") of the Web server. The status code of 200 means that the request was successfully completed by the server, and 1,265 bytes were transferred from the server to `alfonso.usask.ca`.

The access logs provide most of the data needed for workload characterization studies of Web servers. However, they do not provide *all* of the information that is of interest. For example, the log entries tell only the number of bytes transferred for a document, not its *actual size*[2]; there is no record of the *elapsed time* required for a document transfer; and there is no information on the *complete set of files* available on the server, other than those documents that are accessed in the logs. Furthermore, there is no record of whether a file access was human-initiated or software-initiated (e.g., by a *Web crawler*), or what caching mechanisms, if any, are in place at the client and/or the server. These issues are outside the control of our study: our focus is solely on characterizing the workload seen by a typical Internet Web server in its default configuration.

2.5 Performance Issues and Related Work

The overall performance of the World Wide Web is affected by the client, the server, and the capacity of the network links that connect the clients to the server. Efficient Web browsers (clients) can use caching of documents to reduce the loads that they put on Web servers and network links, thereby improving the performance of the Web. A recent study at Boston University [3] studied the effects of client-level caching on Web performance. Several other researchers have studied the use of file caching to reduce network traffic and server loads [4, 8, 10]. Web performance can also be improved by enhancing client-server communication [15, 20].

Although the primary focus of this paper is workload characterization for Web servers, several relevant issues affecting server caching and performance are discussed in Section 5. Client and network performance issues are outside the scope of this paper.

3 Data Collection, Reduction, and Analysis

This section presents an overview of the six separate data sets used in our workload characterization study. Section 3.1 describes the data collection sites, Sections 3.2 and 3.3 present the "raw" log contents, Section 3.4 discusses the reduction of the raw data from the access

[2]These two values can differ for several reasons. For example, there may be overhead bytes associated with the transfer of certain document types, which makes the transfer size reported in the log slightly larger than the actual document size. Furthermore, in most Web browsers, users can abort a document transfer at any time, making the transfer size reported in the log much smaller than the actual document size.

logs into more manageable form, Section 3.5 analyzes document types and sizes, and Section 3.6 summarizes the statistical characteristics of the six data sets.

3.1 Data Collection Sites

The access logs used in this research were obtained from six World Wide Web servers: a department-level Web server at the University of Waterloo (Department of Computer Science); a department-level Web server at the University of Calgary (Department of Computer Science); a campus-wide Web server at the University of Saskatchewan; the Web server at NASA's Kennedy Space Center; the Web server from ClarkNet, a commercial Internet provider in the Baltimore - Washington D.C. region; and the Web server at the National Center for Supercomputing Applications (NCSA) in Urbana-Champaign, Illinois. The Web server for the University of Waterloo's Department of Computer Science is a SUNSparc2, running NCSA httpd version 1.3, and serving a population of 200 graduate students, faculty and staff. The Calgary server is a SUN 4/490, serving about 1300 faculty, staff, and students (graduate and undergraduate). The Web server at the University of Saskatchewan is a Decstation 5000/133, running NCSA httpd version 1.4 for approximately 21,000 students, faculty and staff. The NASA server consists of 4 DEC Alpha 2100 servers on an FDDI ring, each with 128 MB RAM and each running NSCA httpd 1.4. The ClarkNet Web server is a SUNsparc10 with two 60 MHz processors, providing Internet access for 5,000 people (as of August 1995). This machine is running Netscape's Commerce Server 1.1. The NCSA server consists of 8 HP 735 workstations, used in a round-robin fashion to provide Web service [12]. Each workstation has 96 MB RAM and a 130 MB local disk cache. The workstations all run NCSA httpd, and use AFS for file access over an FDDI ring.

3.2 Raw Data

Table 2 summarizes the raw data from the six access logs. For ease of reference, the sites are presented in increasing order of server activity, based on the number of requests per day. The same ordering is maintained in all tables throughout the paper.

The six access logs provide information on servers with very different workloads. Table 2 shows that the Waterloo server had a very light workload, while the Saskatchewan server had an order of magnitude more requests to handle. The ClarkNet and NCSA servers had very heavy workloads, more than an order of magnitude greater than the Saskatchewan server. The level of server activity represented in the six logs varies by almost three orders of magnitude, so that our search for invariants covers light, medium, and heavy workloads. The logs also span different time durations, so that we can study short term, medium term, and long term aspects of Web server file referencing activity.

3.3 Access Log Analysis

The first step in our data analysis was to study the response codes in the Web server access logs. There are many possible responses to client requests. These include: (1) *Successful*: a valid document, which the client has permission to access, was found on the server and returned to the client; (2) *Not Modified*: the client, which already has a copy of the document in its cache but wishes to verify that the document is up-to-date, is told that the document has not been modified at the server (thus no data bytes need to be transferred); (3) *Found*: the requested document is known to reside in a different location than was specified by the URL provided by the client, so the server responds with the new URL (but not the document); and (4) *Unsuccessful*: either no such document exists, the client did not have permission to access this document, or an error occurred (at the server or during network communication).

Table 3 provides an overall view of the response code frequencies observed in the access logs. From Table 3, we can identify the first invariant in Web server traffic. *Successful* responses made up approximately 88% of all responses in the logs.[3] Cache related queries that result in *Not Modified* account for about 8%.

3.4 Data Reduction

Since the *Successful* responses are responsible for all of the documents transferred by the server, only these responses will be used for the remaining analyses in this paper. This simplification provides a reduction in the size of the data sets, and focuses the workload characterization on the most common events.

Table 4 provides a statistical summary of the reduced data sets. This table shows that the number of distinct documents requested from the server is significantly lower than the total number of documents requested, implying that some documents are requested many, many times. The mean size of the documents transferred is quite small (5-21 Kbytes), as might be expected. The table also shows that there is a high degree of variability (measured by the coefficient of variation, CoV) in the transfer size, particularly for the Saskatchewan data set.

3.5 Document Types and Sizes

The high degree of variation in document size is due in part to the wide variety of document types accessed on the server (e.g., HTML, gif, PostScript, audio, MPEG). The next step in our analysis was to classify documents

[3]The Calgary data set is somewhat lower in its successful response rate. Our initial study of invariants began in August 1995 with only three data sets: Waterloo, Saskatchewan, and ClarkNet. Among the three most recent data sets analyzed (Calgary, NASA, and NCSA), only the Calgary data set seems "anomalous" on this invariant. Rather than surreptitiously excluding the Calgary data set from our final paper, we chose to relax our definition of "close enough" for invariants.

Table 2: Summary of Access Log Characteristics (Raw Data)

Item	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
Access Log Duration	8 months	1 year	7 months	2 months	2 weeks	1 week
Access Log Start Date	Jan 1/95	Oct 24/94	Jun 1/95	Jul 1/95	Aug 28/95	Aug 28/95
Access Log Size (MB)	17.9	49.9	222.6	355.8	327.5	267.7
Total Requests	188,636	726,739	2,408,625	3,461,612	3,328,587	2,490,512
Avg Requests/Day	776	2,059	11,255	56,748	237,756	355,787
Total Bytes Transferred (MB)	2,071	7,577	12,343	62,489	27,647	28,268
Avg Bytes/Day (MB)	8.5	21.5	57.7	1,024.4	1,974.8	4,038.3

Table 3: Breakdown of Server Responses for All Data Sets

Response Code	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
Successful	87.8%	78.4%	91.1%	89.6%	88.8%	93.1%
Not Modified	8.2%	13.5%	6.3%	7.7%	8.1%	4.1%
Found	1.6%	4.2%	1.7%	2.1%	0.9%	0.3%
Unsuccessful	2.4%	3.9%	0.9%	0.6%	2.2%	2.5%
Total	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

Table 4: Summary of Access Log Characteristics (Reduced Data)

Item	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
Access Log Duration	8 months	1 year	7 months	2 months	2 weeks	1 week
Access Log Start Date	Jan 1/95	Oct 24/94	Jun 1/95	Jul 1/95	Aug 28/95	Aug 28/95
Access Log Size (MB)	10.4	20.9	143.9	221.2	195.5	172.6
Total Requests	163,112	567,795	2,165,415	3,087,043	2,940,712	2,289,510
Avg Requests/Day	671	1,608	10,119	50,607	210,050	327,073
Distinct Requests	3,406	8,370	18,849	9,355	32,294	23,855
Distinct Requests/Day	14	24	88	153	2,307	3,408
Total Bytes Transferred (MB)	2,071	7,577	12,330	62,483	27,591	28,268
Avg Bytes/Day (MB)	8.5	21.5	57.6	1,024.3	1,970.8	4,038.3
Mean Transfer Size (bytes)	13,313	13,997	5,970	21,224	9,838	12,947
CoV of Transfer Size	3.45	8.01	11.19	3.62	3.84	6.92

by type, using the generic categories HTML, Images, Sound, Video, Formatted, and Dynamic files.

Classification was based on the suffix used in file names. The HTML category represents documents written in the HyperText Markup Language (`.html`, `.htm`, `.shtml`, or `.map`). The Images category represents visual image and graphics files, including Graphics Interchange Format (`.gif`), Joint Photographics Experts Group (`.jpeg`, `.jpg`), bit map (`.xbm`, `.bmp`), and many more. Sound files are audio recordings designated with a suffix of `.au`, `.snd`, `.wav`, `.mid`, `.midi`, `.lha`, or `.aif`. The Video category is for files that are video sequences, such as `.mov`, `.movie`, `.avi`, Quick Time (`.qt`), and Motion Pictures Experts Group (`.mpeg`, `.mpg`) files. The Formatted category includes PostScript (`.ps` or `.eps` suffix) and word processor (`.doc`, `.dvi`) documents. The Dynamic category represents CGI (Common Gateway Interface) scripts (`.cgi`), Perl scripts (`.pl`), and any forms-based or interactive Web documents (e.g., clickable maps, cgi-bin, or CGI strings including the '?' character). All remaining unrecognized document types are classified as Other.

For each of the data sets in Table 4, statistics on the type of document requested were calculated. The results for each log are given in Table 5.

Using Table 5, we can identify a second invariant in Web server workloads. Across the six data sets, HTML and Image documents accounted for 90-100% of the total requests to the server.[4] This observation is consistent with results reported by Sedayao [19] and by Cunha, Bestavros and Crovella [7]. Both of these papers reported that over 90% of client requests were for either HTML or image documents.

Table 5 also indicates that most transferred documents are quite small, which is a third invariant. This phenomenon was also observed by Braun and Claffy [4] for requests to the NCSA's Web server. Despite the fact that Web browsers provide support for the use of multimedia objects like sound and video, documents of these types accounted for only 0.01-1.2% of the requests in the six data sets. However, these types of files account for 0.2-30.8% of the bytes transferred, since these files tend to be much larger than other file types. Future growth in the use of video and audio files, made even easier with tools like CGI scripts and Java, may soon change Invariant 2 and Invariant 3.

The large variation in the mean file sizes for the different document types helps to explain the large coefficient of variation (CoV) values reported in Table 4. The CoV values per document type are much lower in Table 5.

Finally, Table 6 presents a breakdown of the distinct documents requested from each of the servers. Distinct documents are determined by looking at the URL in the access log entries.

[4]In our data sets, there is no invariant for HTML documents alone, or for Image documents alone. In fact, the usage of HTML and Image document types differs dramatically for the Saskatchewan and ClarkNet data sets.

Table 6 illustrates two additional workload invariants. First, only 0.3-2.1% of the requests and 0.4-5.1% of the bytes transferred are for distinct documents. This observation implies that caching documents (at the server, at the client, or within the network) could greatly improve the performance of the server, as has been pointed out by Claffy and Braun [4]. Second, in all six data sets, approximately one-third (e.g., 22.6-42.1%) of all the distinct documents are requested only once, and one-third (e.g., 14.3-42.5%) of the distinct bytes are transferred only once. This observation is somewhat surprising given that the six data sets represent time durations ranging from one week to one year. This "one time" referencing behaviour has obvious implications on the maximum possible effectiveness of document caching policies. Further discussion of these implications is deferred until Section 5.

3.6 Summary

This section has summarized the statistical characteristics of the six data sets used for our workload characterization study. While the six access logs differ greatly in duration and server activity, five workload invariants have been identified. These invariants are summarized in the first five rows of Table 1. The next section presents an in-depth study of file referencing patterns and file size distributions for Internet Web servers, looking for further invariants.

4 Workload Characterization

This section presents a detailed analysis of file referencing behaviours on Internet Web servers, as well as a look at file sizes, the effect of user aborts, and the presence of self-similarity in Web server workloads. We begin with an analysis of file size distributions.

4.1 File Size Distribution

Figure 1 shows the cumulative distribution of the sizes of the distinct documents (files) transferred by each site. While there are a few very small files (< 100 bytes) at each of the sites, most files appear to be in the range of 100 - 100,000 bytes, while a few files ($< 10\%$) are larger than 100,000 bytes. This distribution is consistent with the file size distribution reported by Braun and Claffy [4].

A more rigourous study shows that the observed file size distributions match well with the Pareto distribution [11, 17], for $\alpha < 1$. This observation has been noted in the literature [6, 16], and is confirmed in all six of our data sets. In particular, the tails of the distributions (for file sizes larger than 1024 bytes) are Pareto with $0.40 \leq \alpha \leq 0.63$. This characteristic is present in all six data sets, and is thus added to Table 1.

Table 5: Breakdown of Document Types and Sizes for All Data Sets

Waterloo Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	38.7	50.1	0.01	0.0006	0.3	3.7	7.184
% of Bytes Transferred	35.0	18.9	0.10	0.10	0.2	25.2	20.5
Mean Transfer Size	12,036	4,961	120,973	2,232,051	6,465	90,444	42,130
CoV of Transfer Size	1.82	3.45	1.10	0.00	1.46	1.83	1.89

Calgary Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	47.1	50.3	0.1	0.3	0.04	1.0	1.16
% of Bytes Transferred	13.2	50.2	1.3	11.4	0.01	21.7	2.19
Mean Transfer Size	3,929	13,971	258,196	496,992	4,702	305,444	27,112
CoV of Transfer Size	1.86	3.95	1.49	1.60	1.26	2.77	4.09

Saskatchewan Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	55.6	36.5	0.1	0.004	6.7	0.02	1.076
% of Bytes Transferred	50.7	36.6	1.5	2.6	4.4	0.1	4.1
Mean Transfer Size	5,447	5,980	84,154	3,602,176	3,969	36,055	22,441
CoV of Transfer Size	2.19	2.77	2.62	2.29	2.91	0.08	11.30

NASA Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	30.7	63.5	0.2	1.0	2.6	0.01	1.99
% of Bytes Transferred	18.8	48.1	1.1	29.7	0.3	0.07	1.93
Mean Transfer Size	12,981	16,059	110,311	439,151	2,817	136,436	26,349
CoV of Transfer Size	2.71	2.37	0.80	0.84	0.68	1.85	2.55

ClarkNet Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	19.9	78.0	0.2	0.007	1.2	0.01	0.683
% of Bytes Transferred	15.0	76.6	2.4	2.4	0.8	0.04	2.76
Mean Transfer Size	7,433	9,669	135,082	3,514,759	6,630	36,199	37,138
CoV of Transfer Size	2.14	1.66	1.24	0.35	3.31	1.03	4.25

NCSA Data							
Item	HTML	Images	Sound	Video	Dynamic	Formatted	Other
% of Requests	51.1	48.1	0.2	0.1	0.01	0.006	0.484
% of Bytes Transferred	51.1	36.0	3.5	6.2	0.06	0.2	2.94
Mean Transfer Size	12,950	9,679	197,605	594,796	6,535	369,590	103,783
CoV of Transfer Size	3.56	2.46	5.79	2.18	6.69	2.60	4.38

Table 6: Statistics on Distinct Documents for All Data Sets

Item	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
Distinct Requests/Total Requests	2.1%	1.5%	0.8%	0.3%	1.1%	1.0%
Distinct Bytes/Total Bytes	5.1%	3.8%	2.1%	0.4%	1.5%	2.7%
Distinct Files Accessed Only Once	29.1%	22.6%	42.0%	42.1%	31.9%	35.0%
Distinct Bytes Accessed Only Once	22.8%	19.8%	42.5%	14.3%	24.7%	39.1%

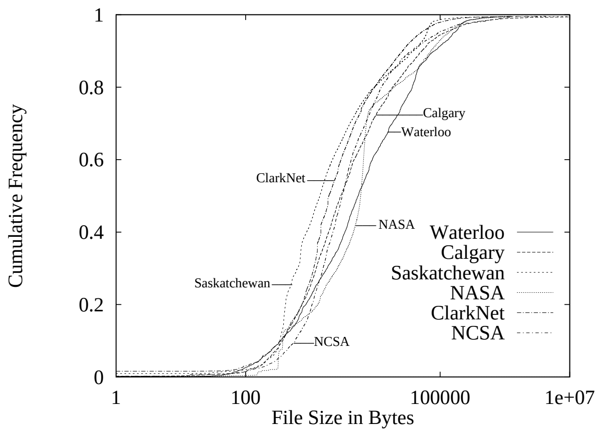


Figure 1: Distribution of File Sizes, by Server

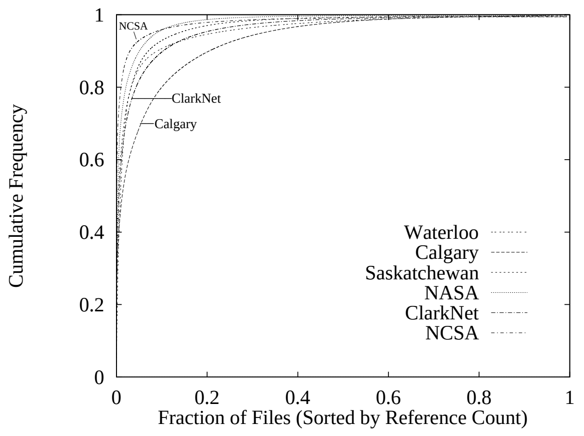


Figure 2: Concentration of References

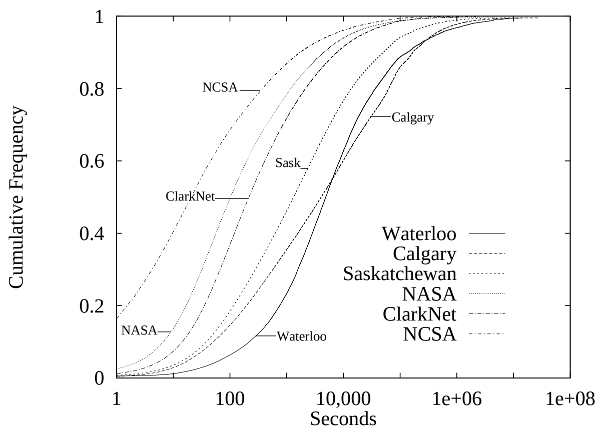


Figure 3: Distribution of Inter-reference Times

4.2 File Referencing Behaviour

This subsection looks at a number of different characteristics in the file referencing patterns at Internet Web servers. The analysis focuses on frequency of reference, concentration of references, temporal locality, inter-reference times, and geographic distribution of references.

4.2.1 Frequency of Reference

Our first analysis focuses on the frequency of reference for different Web documents. Clearly, not all Web documents are created equal. Some are extremely "hot" and popular documents, accessed frequently and at short intervals by many clients at many sites. Other documents are accessed rarely, if at all.

We illustrate this non-uniform referencing behaviour, which we call *concentration*, by sorting the list of distinct files into decreasing order based on how many times they were accessed, and then plotting the cumulative frequency of requests versus the fraction of the total files referenced. The resulting plot for all six data sets is shown in Figure 2.

Figure 2 illustrates the non-uniform pattern of file referencing behaviour: 10% of the distinct documents were responsible for 80-95% of all requests received by the server, at each of the six sites. The NCSA data set shows the most concentration, while the Calgary data set shows the least.

This concentration phenomenon is another invariant in our Web server logs, and is thus added to Table 1. Braun and Claffy have reported similar results for NCSA's Web server in an earlier study [4].

4.2.2 Mean Inter-Reference Times

Our next analysis focuses on the inter-reference time for documents that are accessed more than once. The inter-reference times are computed for each distinct document, and then combined together to form the cumulative distribution of inter-reference times for all documents that are accessed more than once. The cumulative frequency distributions are shown in Figure 3.

Figure 3 clearly illustrates the different workload levels for the six servers. On the lightly-loaded Waterloo server, documents tend to be accessed at long intervals (hours to days). Documents on the busy NCSA server are accessed on a seconds or minutes basis.

A separate statistical analysis (not shown in this paper) indicates that file inter-reference times are exponentially distributed and independent. This observation applies for all six data sets, and is added to Table 1 as an invariant. Clearly, however, the mean inter-reference time depends on the server workload.

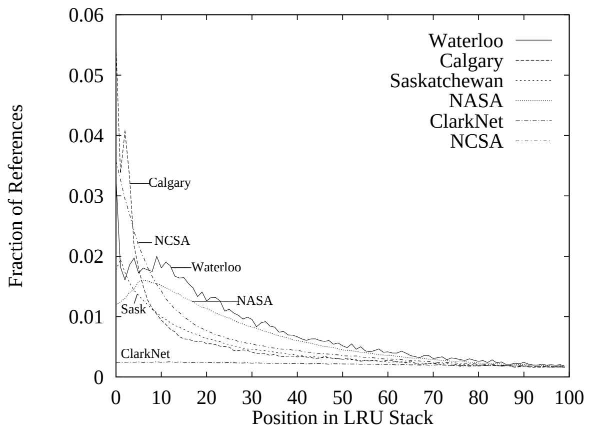


Figure 4: Temporal Locality Characteristics

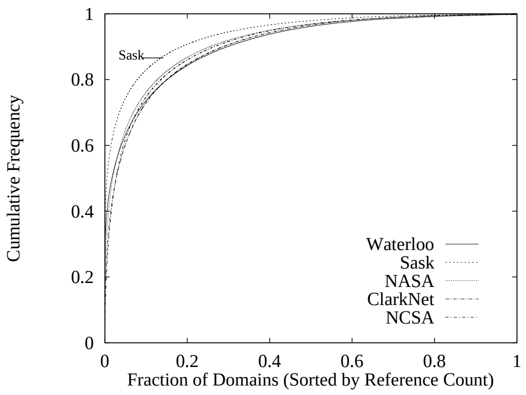


Figure 5: Distribution of References by Domain

4.2.3 Temporal Locality

Access logs were analyzed to look for temporal locality in the file referencing behaviour. Temporal locality refers to the notion of the same document being re-referenced frequently within short intervals.

Temporal locality can be measured using the standard LRU (Least Recently Used) stack-depth analysis. When a document is initially referenced, it is placed on top of the LRU stack (i.e., position 1), pushing other documents down in the stack by one location. When the document is subsequently referenced, its current location in the LRU stack is recorded, and then the document is moved back to the top of the stack, pushing other documents down, as necessary. When the entire log has been processed in this fashion, temporal locality in referencing behaviour is manifested by a high probability of references to locations at or near the top of the LRU stack.

Figure 4 shows the results of our LRU stack-depth analysis for all six data sets. The Calgary data set shows the highest degree of temporal locality, while the ClarkNet data set shows the least. There is thus no invariant evident in these data sets for temporal locality.

Our speculation is that the level of multiplexing in a busy Web server is large enough to mask any evidence of temporal locality in the access logs. Client-side caching mechanisms may also serve to remove temporal locality from the reference stream seen at the server, as has been shown in other client-server environments [9].

4.2.4 Geographic Distribution

Our final analysis of file referencing behaviour examines the geographic distribution of document requests. This analysis makes use of the IP addresses of the requesting hosts in the access log. In particular, the network component of the IP address (based on a Class A, Class B, or Class C address) is used to determine if a requesting host is *local* or *remote* relative to the Web server. The network identifier in each IP address is further used to classify requesting hosts into *domains* (not to be confused with *domain names*) that have the same network address.

Table 7 shows the geographic distribution of requests and bytes transferred at the six sites. For example, 77.7% of all the requests to the Waterloo server came from remote hosts, while local hosts generated the remaining 22.3% of the requests. In terms of bytes transferred, 81.7% of the requested bytes were transferred to remote hosts, with 18.3% to local hosts. The rest of the table is organized in a similar manner.

On all six Web servers, remote hosts send the most requests and receive the most data. Remote hosts account for over 75% of requests on all but one server (Calgary), and well over half of the total bytes transferred on all servers. This observation is reported in Table 1 as another invariant.

The local access patterns at the Saskatchewan and Waterloo servers are quite similar. The similarity is likely caused by the use of the Web in teaching and research activities. The access pattern at NCSA, NASA, and ClarkNet is substantially different, with remote accesses accounting for almost all of the requests and transferred data. The likely explanation for this behaviour is that there are very few "local" hosts for these organizations.

Figure 5 shows the distribution of references by the number of domains accessing the Web server.[5] A small number of domains account for a significant portion of the requests, while the remaining requests are received from several thousand domains. In all six data sets, 10% of the domains accounted for at least 75% of the requests (Invariant 10 in Table 1).

[5]The Calgary data set is not shown since the "sanitized" logs that we received from the University of Calgary did not show host names or IP addresses for each request, but only a boolean indicator of LOCAL or REMOTE.

Table 7: Geographic Distribution of Requests for All Data Sets

Local Hosts						
Item	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
% All Requests	22.3	46.4	24.9	6.3	1.9	1.2
% All Bytes	18.3	36.6	24.8	2.7	1.6	0.5

Remote Hosts						
Item	Waterloo	Calgary	Saskatchewan	NASA	ClarkNet	NCSA
% All Requests	77.7	53.6	75.1	93.7	98.1	98.8
% All Bytes	81.7	63.4	75.2	97.3	98.4	99.5

4.3 Self-Similarity

Recent work has suggested that World Wide Web traffic may be self-similar [6]. This section briefly describes the tests that were performed to check for self-similarity in Web server workloads. In short, we found a slight degree of self-similarity (a Hurst parameter value of $H \approx 0.65$) in the ClarkNet data set, very little ($H \approx 0.53$) in the Saskatchewan data set, and none at all in the Waterloo data set. Only the analysis of the ClarkNet data set is described here.

One aspect of self-similarity is the absence of a characteristic size of a traffic burst. To assess this effect, the server workload data (i.e., bytes transferred per unit of time) were plotted and inspected visually, as in Leland *et al* [13]. The results are shown in Figure 6, for four different time scales. The topmost graph (1000 second intervals) shows the full week of ClarkNet data, in which a distinct daily usage pattern is seen. The bottom graph shows the workload per 1 second interval, which is the resolution of the access log timestamps. The horizontally shaded regions in the upper plots show the portion of the trace that is expanded in the next lower time scale.

Moving from the bottom plot to the top plot in Figure 6, burstiness clearly exists across several different time scales. The results suggest that there is some evidence of self-similarity in this workload.[6]

A more rigourous analysis confirms this intuition. The analysis, as described in [13], makes use of three statistical tests: autocorrelation, variance-time plot, and R/S analysis. These tests were performed for both one hour and eight hour periods of the data.

Figure 7 summarizes the results for one eight-hour sample. The autocorrelation plot in Figure 7(a) shows that the autocorrelation coefficients are small, but nonzero, even at large lags. This observation is more evident in the detailed plot (Figure 7(b)), providing evidence of a slowly decaying autocorrelation function, which is one of the manifestations of self-similarity. Figure 7(c) shows a variance-time plot with a slope deviating significantly from -1 (the dashed line), again suggesting self-similarity. Finally, Figure 7(d) shows an R/S pox diagram. This scatterplot appears to have a

[6] We make this claim of self-similarity even though the presence of stationarity at the highest time scale is clearly questionable.

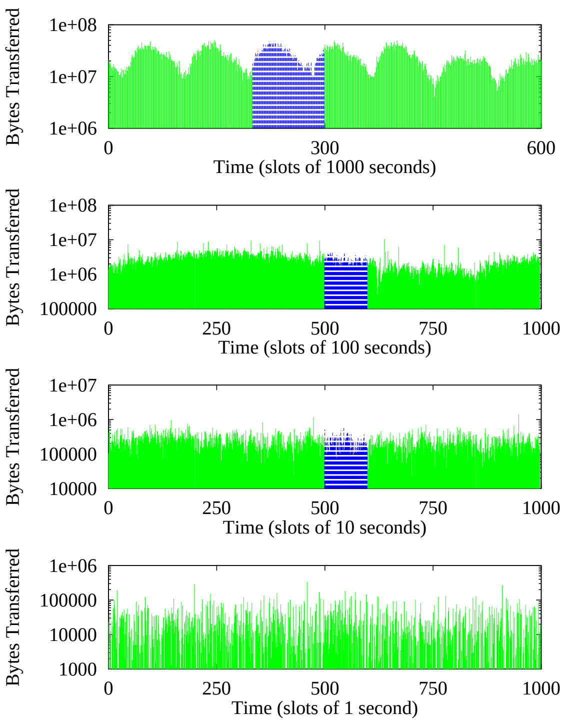


Figure 6: Server Workload at Four Different Time Scales (ClarkNet Data)

slope greater than 0.5 (the lower dashed line), and less than 1.0 (the upper dashed line). Linear regression on the R/S plot resulted in a Hurst parameter value of $H = 0.65$, suggesting a small degree of self-similarity in the ClarkNet server workload.

Self-similarity does not appear to be an invariant in all Web server workloads, though it does appear to be a property when Web traffic is heavy, as reported in [6].

4.4 Aborted Connections

Several Web documents appeared in an access log multiple times, with the same URL each time, but with different transfer sizes at different points in the log. There are two possible causes for these "anomalies" in the Web server access logs. First, a user may edit and physically change a Web document at any time. Second, a Web client may abort a connection in the midst of a document transfer (i.e., the user clicks on the "Stop" button in the Web browser).

From the standpoint of access log analysis, aborts appear the same as a file modification because the access logs record the number of bytes transferred by the server, not the actual size of the file being transferred. However, aborted connections *are* reported in the Web server error log. We obtained the error log for the Saskatchewan data set, but not for the other five data sets in our study. With this single error log, it is possible to assess the impact of user aborts on the results reported in this paper.

An analysis was thus performed on file "modification" times, where a modification refers to a document that changes size between references to that document. Figure 8 shows the cumulative distribution of the elapsed time between such modifications, aggregated over all files that experienced modification. Most files tend to go unmodified for extended periods of time. The small number of files that are modified only a few seconds after the previous modification are likely the result of a user creating or modifying a Web page, viewing the page, then editing and re-viewing the page until the desired modifications are complete.

The Saskatchewan* curve in Figure 8 shows the file modification time distribution for the Saskatchewan data set once all aborted connections (1.1% of all requests) have been removed. The resulting distribution is significantly flatter, showing that files are actually modified much less frequently than the raw data in the log suggests. Removing aborted connections from the other five logs (if the error logs were available) would likely show similar results.

Table 8 summarizes information about aborted connections in the Saskatchewan data set. While the number of aborted connections is quite low, the number of bytes transferred by aborted connections is somewhat larger. Furthermore, remote users are more likely to abort a connection than are local users, as expected.

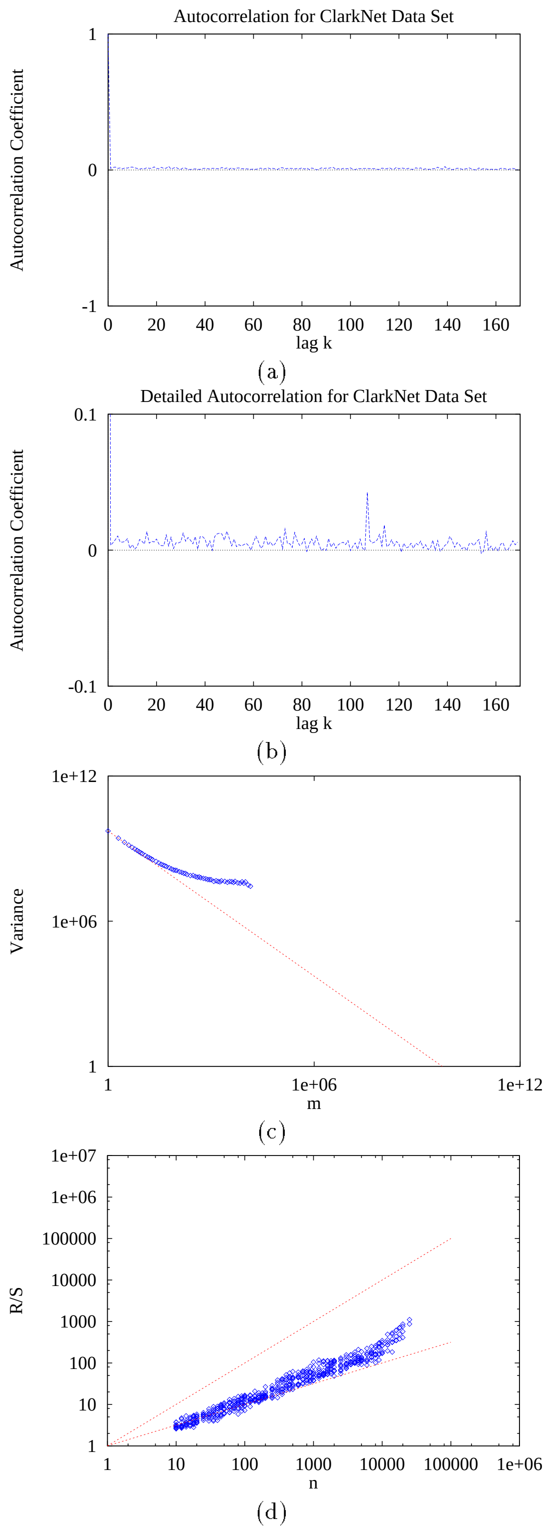


Figure 7: Auto-Correlation, Variance-Time and R/S Plots (ClarkNet Data)

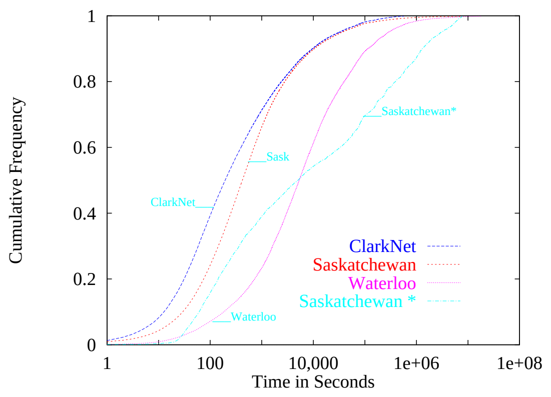


Figure 8: Cumulative Distribution of Detected File Modification Times

Table 8: Aborted Connections (Saskatchewan Data)

Item	Local	Remote	All
% of Connections Aborted	0.9	1.2	1.1
% of Bytes Transferred	4.0	5.7	5.1

4.5 Summary

This section has presented a detailed study of Web server workload characteristics. Results were presented for file size distributions, file referencing patterns, aborted connections, and self-similarity in Web server workloads.

From the analyses reported in this section, five additional workload invariants have been identified. These invariants appear in the last five rows of Table 1.

5 Performance Implications

We conclude our paper with a discussion of caching and performance issues for Internet Web servers. Despite the low temporal locality seen in most Web server workloads, caching still appears to be a promising approach to improving Web performance because of the large number of references to a small number of documents (Invariant 4 from Table 1), the concentration of references within these documents (Invariant 7), and the small average size of these documents (Invariant 3). We intentionally leave unspecified the location of the cache[7] and the size of the cache, focusing instead on the use of our workload invariants to estimate the maximum performance improvement possible with Web server caching. For simplicity, the discussion assumes that all Web documents are read-only (i.e., never modified), and that file-level (not block-level) caching is used. Misses due to "cold start" are also ignored.

[7] Several logical choices exist: (1) at the client, or the client's network, to reduce requests to a remote server; (2) at the server, or the server's network, to reduce disk accesses and/or byte transfers on the server's network; (3) in the network itself, to reduce repeated "pulls" of the same document across a geographic region of the network; and (4) a combination of the above.

5.1 A Basic Tradeoff: Requests versus Bytes Transferred

There are two main elements that affect the performance of a Web server: the number of requests that a server must process, and the number of data bytes that the server must transfer (i.e., disk I/O's, packets).

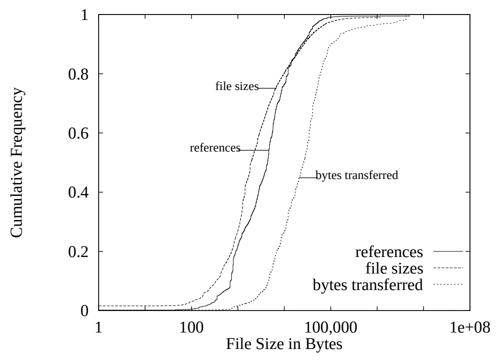
There is thus a choice to be made between caching designs that reduce the number of requests presented to Internet Web servers, and caching designs that reduce the volume of network traffic[8]. Both approaches represent possible avenues for improving Web server performance, but optimizing one criterion does not necessarily optimize the other. The choice between the two depends on which resource is the bottleneck: CPU cycles at the server, or network bandwidth.

We illustrate this tradeoff in Figure 9. While the discussion here focuses only on the ClarkNet data set, similar observations apply for the other data sets.

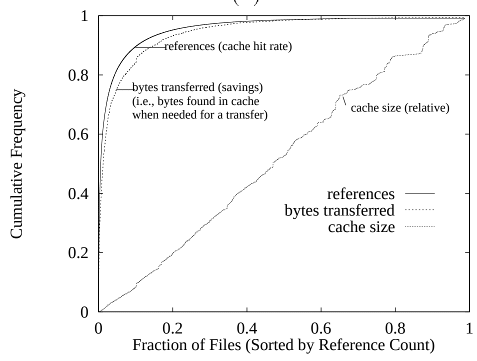
The topmost graph (Figure 9(a)) illustrates the relationship between the size of files on a Web server (from Figure 1), the number of references to those files, and the number of data bytes that references to those files generate (i.e., the "weighted value" obtained from the product of file size and number of times that a file is referenced). This graph shows that 80% of all the documents requested from the ClarkNet server were less than 10,000 bytes in size. 76% of all references to the server were for files in this category. Thus, caching a large number of small files would allow the server to handle most of the requests in a very efficient manner. However, Figure 9 also points out that the references to files less than 10,000 bytes in size generate only 26% of the data bytes transferred by the server. Furthermore, looking at the tail of the distribution, documents over 100,000 bytes in size are responsible for 11% of the bytes transferred by the server, even though less than 0.5% of the references are to files in this category (Invariant 6). What this means is that in order to reduce the number of bytes transferred by the server as much as possible, a few large(r) files would have to be cached. That is, the server must sacrifice on "cache hits" for many small requests in order to save on bytes transferred for large requests.

The remaining two plots in Figure 9 illustrate the tradeoff in more detail. The middle plot (Figure 9(b)) shows the results for a cache designed to maximize cache hits for requests (i.e., to reduce the number of requests to the server). In this graph, the top line represents the cache hit rate for requests, the bottom line represents the cache size, and the middle line represents the potential savings in bytes transferred by the server when the cache is present. In this design, for example, caching

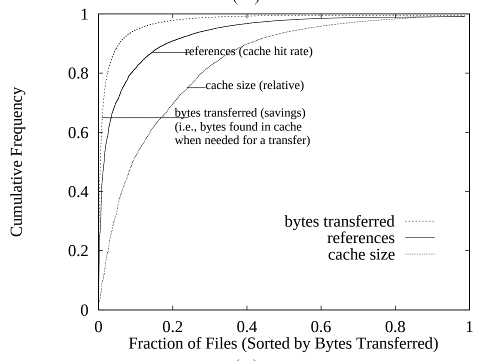
[8] Clearly, reducing the number of requests also reduces the volume of network traffic, but the main focus of the two approaches is different, as will be shown.



(a)



(b)



(c)

Figure 9: Comparison of Caching and Performance Issues for ClarkNet Data

10% of the server's distinct files (namely, the most frequently accessed documents) for the ClarkNet data set results in a cache hit rate of 90% (the top line in the graph). The documents in the cache, which represent the potential savings in bytes transferred, account for 84% (the middle line in the graph) of the bytes transferred by the server, and the cache size would need to hold 8.3% (the bottom line in the graph) of the total distinct bytes referenced in the server access log.

The bottom plot (Figure 9(c)) represents the results for a cache designed to reduce bytes transferred. In this graph, the top line represents the savings in bytes transferred, the bottom line represents the cache size, and the middle line represents the cache hit rate. In this design, for example, caching 10% of the server's files (namely, the 10% of the documents that account for the most bytes transferred) results in an 82% cache hit rate (the middle line). The documents in the cache would account for 95% (the top line) of the bytes transferred, but the cache would have to be large enough to contain 52% (the bottom line) of the distinct bytes represented in the server access log. Clearly there is a tradeoff to be made in cache size, cache hit rate, number of server requests, and number of bytes transferred by the server.

5.2 Other Issues

Our final comments concern "one timers", cache replacement strategies, and thresholding approaches to cache management. We are currently investigating these caching issues using our Web server workloads.

First, the "one time" referencing (Invariant 5) of Web documents is a concern.[9] While this effect could be simply an artifact of the finite duration of the access logs studied, or something as innocent as the deletion or renaming of Web documents, the effect is present across *all* access log time durations studied. This one-time referencing behaviour means that, on average, one-third of a server cache could be cluttered with useless files. Techniques to expunge such files from a cache, such as timeouts on cached files, are desirable. Invariant 8 may be useful in setting proper timeout values for documents in the cache. Another approach would be to cache only on the *second* reference to a file within a specified time period, rather than the first. However, the merits of this approach seem dubious, since keeping state information about some documents that are not yet in the cache would incur almost as much overhead as caching the document in the first place.

Second, the fact that temporal locality was *not* present in all data sets suggests that LRU as a cache replacement policy may not work well for *all* servers. Policies such as Least Frequently Used (LFU) may be more attractive because of the concentration of references (Invariant 7), and also because LFU easily deals with one-timers. Our trace-driven simulations to date do indeed

[9]The advent of *Web crawlers* may change Invariant 5 to be "N timers", for some small integer N. However, the argument that we make here still applies.

show that LFU is consistently superior to LRU, with FBR (Frequency Based Replacement) providing performance in between that of LFU and LRU. Adding caching *partitions* based on document types helps in most cases, but does not change the relative ordering of the policies studied.

Third, there may be merit in using "size thresholds" in cache management, to better cope with the "heavy tailed" Pareto distribution of file sizes (Invariant 6), and the issues raised in Section 5.1. For example, two such threshold policies might be "never cache a document larger than X bytes" (because it uses up too much cache space, and adversely impacts hit rate), or "never cache a document smaller than Y bytes" (because it does not save much on bytes transferred by the server). Trace-driven simulation experiments confirm this intuition. While the performance gains are small, caching policies with a maximum document size threshold only seem to work best.

Finally, as a small but practical matter, Web servers should avoid doing name lookups for *each* incoming client request when producing the access log, particularly when successive requests are from the same requesting host. That is, servers should exploit whatever temporal locality exists in the incoming reference stream of requesting hosts (not analyzed in this paper) to avoid the (slow) name lookup whenever possible. A small cache (e.g., 20 entries) of the results of recent name lookups should suffice (e.g., 75% hit rate). This small refinement alone may help improve response times for heavily loaded Web servers. Turning off the name server lookup feature is another option.

6 Conclusions

This paper has presented a detailed workload characterization study for Internet World Wide Web servers. The study used logs of Web server accesses at six different sites: three from university environments, two from scientific research organizations, and one from a commercial Internet provider. The logs represent three different orders of magnitude in server activity, and span two different orders of magnitude in time duration.

From these logs, we have been able to identify ten invariants in Web server workloads. These invariants were summarized in Table 1 at the start of the paper. These invariants are deemed important since they potentially represent universal truths for all Internet Web servers.

The invariants were used to identify two possible strategies for the design of a caching system to improve Web server performance, and to determine bounds on the performance improvement possible with each strategy. The performance study identified the distinct tradeoff between caching designs that reduce network traffic, and caching designs that reduce the number of requests presented to Internet Web servers. While the two approaches are somewhat at odds with each other, both represent possible avenues for improving Web server performance.

Several relatively recent Web forces may someday undermine or change our ten Web server invariants. These forces include Web crawlers, improved protocols for Web interaction, small-scale and large-scale Web caching architectures, and a growing trend toward the use of video, audio, and interactivity on the Web (e.g., CGI, Java). It will be interesting to see how long our invariants remain invariant.

Acknowledgements

The authors are grateful to the following people for making their Web server access logs available for our study: Jamie Hodge, Department of Computer Science, University of Waterloo; Robert Fridman, Department of Computer Science, University of Calgary; Earl Fogel, Department of Computing Services, University of Saskatchewan; Jim Dumoulin, NASA (Kennedy Space Center); Stephen Balbach, ClarkNet; and Robert McGrath, NCSA.

Funding for this research was provided by NSERC Research Grant OGP0120969, and by an NSERC Postgraduate Scholarship. Part of this work utilized the Mass Storage System at the National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign.

The authors thank Rick Bunt (U of S), John Dilley (HP), and the anonymous referees for their constructive comments and suggestions, which helped to improve the clarity of the final paper.

This research was motivated by a discussion with Vern Paxson in May 1995. Vern Paxson was also instrumental in establishing the Internet Traffic Archive.

References

[1] M. Andreessen, "NCSA Mosaic Technical Summary", National Center for Supercomputing Applications, 1993.

[2] T. Berners-Lee, L. Masinter and M. McCahill, "Uniform Resource Locators", RFC 1738, December 1994.

[3] A. Bestavros, R. Carter, M. Crovella, C. Cunha, A. Heddaya and S. Mirdad, "Application-Level Document Caching in the Internet", *Proceedings of the Second International Workshop on Services in Distributed and Networked Environments (SDNE '95)*, Whistler, BC, Canada, pp. 166-173, June 1995.

[4] H. Braun and K. Claffy, "Web Traffic Characterization: An Assessment of the Impact of Caching Documents from NCSA's Web Server", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[5] R. Cáceres, P. Danzig, S. Jamin and D. Mitzel, "Characteristics of Wide-Area TCP/IP Conversations", *Proceedings of ACM SIGCOMM '91*, Zürich, Switzerland, pp. 101-112, September 1991.

[6] M. Crovella and A. Bestavros, "Explaining World Wide Web Traffic Self-Similarity", *Proceedings of the 1996 ACM SIGMETRICS Conference*, Philadelphia, PA, May 1996.

[7] C. Cunha, A. Bestavros and M. Crovella, "Characteristics of WWW Client-Based Traces", Technical Report BU-CS-95-010, Boston University Computer Science Department, 1995.

[8] P. Danzig, M. Schwartz and R. Hall, "A Case for Caching File Objects Inside Internetworks", *Proceedings of ACM SIGCOMM '93*, San Francisco, California, pp. 239-248, September 1993.

[9] K. Froese and R. Bunt, "The Effect of Client Caching on File Server Workloads", *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, January 1996.

[10] S. Glassman, "A Caching Relay for the World Wide Web", *First International Conference on the World Wide Web*, Geneva, Switzerland, May 1994.

[11] N. Johnson and S. Kotz, Editors-in-Chief, *Encyclopedia of Statistical Sciences, Volumes 6 and 9*, John Wiley & Sons, Inc., New York, 1988.

[12] T. Kwan, R. McGrath, and D. Reed, "NCSA's World Wide Web Server: Design and Performance", *IEEE Computer*, Vol. 28, No. 11, pp. 68-74, November 1995.

[13] 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.

[14] NSFNET Statistics. Data available by anonymous ftp from nic.merit.edu/statistics/nsfnet.

[15] V. Padmanabhan and J. Mogul, "Improving HTTP Latency", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[16] V. Paxson, "Empirically-Derived Analytic Models of Wide-Area TCP Connections", *IEEE/ACM Transactions on Networking*, Vol. 2, No. 4, pp. 316-336, August 1994.

[17] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling", *Proceedings of ACM SIGCOMM '94* London, England, pp. 257-268, August 1994.

[18] V. Paxson, "Growth Trends in Wide Area TCP Connections", *IEEE Network*, Vol. 8, No. 4, pp. 8-17, July/August 1994.

[19] J. Sedayao, "Mosaic Will Kill My Network!", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[20] M. Spasojevic, M. Bowman and A. Spector, "Using a Wide-Area File System Within the World-Wide Web", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[21] A. Tanenbaum, *Computer Networks*, Second Edition, Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[22] World Wide Web Frequently Asked Questions, URL: http://www.io.org/faq/www/index.html

For More Information

Martin Arlitt's M.Sc. thesis will be available via URL `http://www.cs.usask.ca/projects/discus/` in May 1996. The C programs used in this study to process Web server logs will be available on an "as is" basis from the same site. We hope to make one or more of our Web server access logs available to other researchers via the Internet Traffic Archive (ITA), located at URL `http://town.hall.org/Archives/pub/ITA/`