Traffic Analysis of a Web Proxy Caching Hierarchy

Anirban Mahanti Carey Williamson Derek Eager
Department of Computer Science
University of Saskatchewan
Canada S7N 5A9
email: {anm474, carey, eager}@cs.usask.ca

Appears in IEEE Network Magazine: Special Issue on Web Performance, May/June 2000

Abstract

Understanding WWW traffic characteristics is key to improving the performance and scalability of the Web. In this paper, Web proxy workloads from different levels of a caching hierarchy are used to understand how the workload characteristics change across different levels of a caching hierarchy. The main observations of this study are: HTML and image documents account for 95% of the documents seen in the workload; the distribution of transfer sizes of documents is heavy-tailed, with the tails becoming heavier as one moves up the caching hierarchy ; the popularity profile of documents does not precisely follow the Zipf distribution; one-timers account for approximately 70% of the documents referenced; concentration of references is less at proxy caches than at servers, and concentration of references diminishes as one moves up the caching hierarchy; and the modification rate is higher at higher-level proxies.

Keywords: World-Wide Web, HTTP Traffic Characterization, Web Proxy, Caching Hierarchy

1 Introduction

The World-Wide Web (WWW or Web) has experienced phenomenal growth in recent years. This growth of the Web has contributed significantly to the network traffic on the Internet, and motivated much research into improving the performance and scalability of the Web.

In recent years, Web proxy caches have been deployed to reduce network traffic and provide better response time for Web accesses. A Web proxy consists of application level software that accepts document retrieval requests from a set of clients, forwards these requests to appropriate servers if the requested documents are not already present in the proxy's cache, and sends documents back to the clients. Originally, proxies were designed to allow network administrators to be able to control Internet access from within an Intranet [1]. It was recognized, however, that proxies may also serve as repositories for frequently requested documents. This role of proxies has made them very popular. Caching documents at the proxy can save network bandwidth and reduce network latency for document accesses [2].

Most browsers can be configured to use a proxy. Proxies can be deployed almost anywhere in the Internet. A second-level cache, the first-level being the browser cache, can be provided by a proxy cache that serves requests from a large community such as a large corporation, a university, or the customers of an Internet Service Provider. Higher-level proxies have also been deployed. These proxies typically have other second-level caches (i.e., lower-level proxies) as their clients.

Web caches connected in the above described tree-like fashion are said to form a cache hierarchy. Web caches distribute load away from server "hot spots", saving wide-area network bandwidth, and reducing access latencies. Lower-level caches typically configure a higher-level cache as their parent. The first Web proxy, the CERN httpd server, allowed caches to be arranged hierarchically. In such Web caches, requests that cannot be satisfied from a lower-level cache are forwarded to the higher-level cache, with the expectation that some of these requested documents will be stored there. The Internet Cache Protocol (ICP) [4] was developed as a part of the Harvest caching project [5] to provide a more efficient method of inter-cache communication. ICP allows lower-level caches to send queries to a higher-level cache enquiring whether or not it has a copy of the requested document. Requests are forwarded to the higher-level only if a copy of the document is found; otherwise, the request is forwarded to the origin server.

Understanding WWW traffic characteristics is key to the design of techniques that save network bandwidth, reduce latency, and improve response time of Web accesses. This paper is concerned with understanding factors that can affect the performance of Web proxy caches. The study builds upon previous work done by Arlitt and Williamson [6] on Web server workloads, Cunha *et al.* [7] on Web client workloads, and Abdulla *et al.* [8] on Web proxy workloads. The present work differs from that in [8] as workloads from proxies at different levels of a caching hierarchy are considered, permitting a more complete study of how the workload characteristics change as one moves from the client side to the server side of the network.

The rest of the paper is organized as follows. Section 2 describes the data collection, reduction, and analysis of access logs that was performed. Section 3 presents the results of the workload analysis, followed by conclusions in Section 4.

2 Data Collection, Reduction, and Analysis

Our traffic analysis is conducted using access logs from Web proxy servers. Each entry in the access logs records the URL of the document being requested, the date and time of the request, the name (or the IP address) of the client host making the request, the number of bytes returned to the requesting client, and additional information that describe how the client's request was treated at the proxy. Processing these log entries can produce useful summary statistics about workload volume, document types and sizes, popularity of documents, and proxy cache performance. Section 2.1 describes the data collection sites and the "raw" data sets, while Section 2.2 describes the reduction of the raw data sets to only contain useful information for the analysis performed in this paper.

2.1 Configuration of Proxy Sites

The access logs for this study were obtained from three World-Wide Web proxy servers, namely: the proxy server at the University of Saskatchewan; the CANARIE proxy [9] located at Bell Canada in Toronto; and the NLANR proxy [10] at the University of Illinois, Urbana-Champaign. Each of these sites continuously records access logs, which were obtained on a daily basis using FTP. Further detail on each of these sites is provided below.

The Web proxy server at the University of Saskatchewan represents an institution-level Web proxy in the CA*net II caching hierarchy. It serves several hundred users on the University of Saskatchewan campus who have configured their browsers to use the proxy cache. This proxy server is operated by the Department of Computing Services at the University of Saskatchewan. The proxy server uses a Digital AlphaServer 1200 5/400 with two 400 MHz processors running `Squid` version 1.2.beta20 with 5 GB disk cache. The proxy is configured to use the CANARIE cache as its parent. That is, cache misses at the University of Saskatchewan cache result in requests to the CANARIE cache for the missing document.

The CANARIE proxy cache is the root of the CA*net II caching hierarchy. This proxy is a SUN Ultra Sparc 180 MHz machine running `Squid` version 1.2.beta22. The CANARIE proxy server is physically located at Bell Canada, Toronto, though it is administratively controlled by the CANARIE ARDNOC (Advanced Research and Development Network Operations Centre) in Ottawa. This cache currently functions as a parent for several first-level proxies, including proxies at the University of Saskatchewan, the University of Alberta, Dalhousie University, and McMaster University. There are also a small number of users who configure their browsers to directly use the CANARIE proxy cache. The CANARIE proxy has parent links to two nodes in the NLANR (National Laboratory for Applied Networking Research) caching hierarchy, namely the Pittsburgh NLANR node, and the NLANR node at the University of Illinois, Urbana-Champaign (UIUC).

The NLANR traces in this study come from the NCSA (National Center for Supercomputing Applications) proxy server at the University of Illinois, Urbana-Champaign (UIUC). This site represents one of several top-level nodes in the NLANR Web caching hierarchy; it receives requests from sibling caches at the top level, as well as from lower-level caches that use it as a parent. The NCSA proxy server is a Digital AlphaServer 1000 266 MHz machine running `Squid` version 1.2.beta17 with about 10GB of disk cache.

The access logs of individual days were concatenated to obtain longer data sets for each site. A 45-day trace was created for the University of Saskatchewan proxy server by concatenating access logs from February 5 to March 30, 1999 [1]. Similarly, a 45-day CANARIE trace spanning February 2 to March 24, 1999 [2], and a 30-day NLANR trace spanning February 3 to March 4, 1999 were created. The University of Saskatchewan proxy server recorded a total of 30,330,149 requests in 45 days of activity. The CANARIE proxy server recorded a total of 20,725,429 requests in 45 days of activity. The NLANR access logs recorded 35,631,782 requests in the 30 days of activity. These three data sets will be referred to as USask, CANARIE, and NLANR in the rest of this paper. The access logs are presented in this relative order to reflect the progression from an institution-level Web proxy cache to a top-level Web proxy cache.

[1] The access logs for 9 days were not available, due to downtime for server upgrades and network outages.

[2] The access logs for 6 days were not available.

2.2 Data Reduction and Analysis

The extremely large volume of the raw proxy traces necessitated pruning of the data sets to contain only the information useful for our study. Both the USask and the CANARIE Web proxies were configured to use ICP for inter-cache queries. ICP queries account for a significant fraction of the total requests in the proxy traces, though they do not result in document transfers. An analysis of the raw data sets showed that 17% of the requests made to the USask proxy were ICP queries. Similarly, 52% of the requests in the CANARIE proxy trace were ICP queries, while for the NLANR proxy, none of the requests were ICP queries, since the NLANR access log was not configured to record activity at the ICP port.

After removing the ICP queries from the raw data sets, the next step used the HTTP reply codes in the access logs for data reduction. The HTTP reply codes describe how a client's request was serviced. For example, a reply code of 200 (OK) means that a valid document was made available to the client, either directly from the proxy cache, or by retrieving the document from another proxy cache or from the origin server. A reply code of 304 (Not Modified) implies that a client had issued a GET If-Modified-Since request to determine whether or not the client's cached copy of a document was up-to-date, and the server (or some intermediate proxy) replied indicating that the client has a valid copy of the document. An HTTP response code of 206 (Partial Content) implies partial transfer of the document to the client in response to a partial GET request. Partial GETs are used to recover efficiently from partial failed transfers in HTTP/1.1. An HTTP response of 302 (Found) means that the requested document is known to reside in a different location than that specified by the URL.

In the data sets, approximately 90% of the HTTP requests made to the proxy result in the client successfully obtaining the document (HTTP 200 and HTTP 304). In this study, we consider all requests that would result in the document being accessed from the origin server in the absence of intermediate proxies. Therefore, only the 200 (OK) and 206 (Partial Content) status-codes are considered.

Table 1 summarizes the reduced access logs for the three Web proxy sites. Based on the average number of requests seen per day at each proxy server, the NLANR proxy server has the highest activity, while the CANARIE proxy server has the least activity. This is not surprising since very few institutional caches currently use the CANARIE proxy cache[3].

Table 1 also indicates the number of distinct documents, servers, and clients recorded in the access logs. Each proxy handles requests for millions of distinct documents located at thousands of different Web servers. The number of clients (i.e., distinct client IP addresses seen) varies quite significantly in the three traces. This number is not known precisely for the NLANR log, since the client IP addresses in the NLANR access logs are randomized every day (for privacy concerns). However, on any particular day, about 700 clients generate requests to the NLANR cache.

Overall, the mean and median document transfer sizes are quite small, as has been reported in previous studies of Web servers [6, 11] and Web proxy servers [8]. In these data sets, the mean size of the documents transferred ranges from 8-13 KB, while the median is in the range of 2-3 KB. The mean transfer size is larger than the median transfer size because there are some very large documents that skew the mean of the transfer size distribution. There is also high variability in the sizes of documents transferred, as indicated by the coefficient of variation

[3] The majority of the requests at the CANARIE proxy are from the USask proxy server.

Table 1: Summary of Web Proxy Access Log Characteristics (Reduced Data)

Item	USask	CANARIE	NLANR
Total Requests	21,070,330	7,310,038	24,560,611
Avg Requests/Day	468,229	162,445	818,687
Total Bytes Transferred (GB)	177	89	345
Avg Bytes/Day (MB)	3,943	1,954	11,516
Distinct Documents	5,510,561	4,571,539	8,482,661
Distinct Servers	133,692	117,433	272,509
Distinct Clients	1,117	13	-
Mean Transfer Size (Bytes)	8,422	12,297	14,066
Median Transfer Size (Bytes)	2,500	3,455	3,128
Coefficient of Variation	13.79	13.28	17.60

(COV) reported in Table 1. These reduced data sets are used in the analyses in the rest of the paper.

3 Proxy Traffic Analysis

The following sections present the results from a detailed traffic analysis of Web proxy workloads. The following characteristics are studied: document types and sizes, proportion of documents accessed only once in the access logs, distribution of transfer sizes, popularity of documents, and rate of document modifications.

3.1 Document Types and Sizes

The next step in the analysis classifies document requests (based on URL extensions[4]) into the following categories:

- HTML: ".html",".shtml", ".htm", and ".map".
- Image: ".gif", ".tiff", ".jpeg", ".xwd", ".rgb", ".xbm", ".tif", ".gif89", ".bmp", ".ief", ".jpe", ".ras", ".pgm", ".ppm", ".pcx", ".pic" and ".jpg".
- Audio: ".au", ".aiff", ".aif", ".aifc", ".mid", ".snd", ".wav" and ".lha".
- Video: ".mov", ".qt", ".avi", ".mpe", ".movie", ".mpeg", ".mpg", ".mp2" and ".mp3".
- Compressed: ".z", ".gz", ".zip" and ".zoo".
- Formatted: ".ps", ".pdf", ".dvi", ".ppt", ".tex", ".rtf", ".src", ".doc" and ".wsrc".

[4] Note that all documents with a "cgi-bin" or "?" in the URL string are classified as Dynamic documents. The content type information available in the access logs is used only if URL extensions do not classify the documents into one of the above specified generic types.

- Dynamic: ".cgi", ".pl" and ".perl".

Any document that could not be classified under one of the above categories was placed in the Others category.

The results of this analysis for the three data sets are summarized in Table 2. The table shows that HTML and image documents account for close to 95% of the total requests. Similar results were reported for client traces by Cunha *et al.* [7], and for server traces by Arlitt and Williamson [6].

Unlike Web server workloads, however, Table 2 shows that documents of image type are consistently the most requested document type (68-78%), followed by HTML documents (about 20%). Similar observations were made by Abdulla *et al.* [8] in their characterization of Web proxy traffic. It is also observed that image type documents are responsible for the highest percentage of bytes transferred (40-52%), followed by HTML documents (18-23%).

Table 2: Breakdown of Document Types and Sizes

USask								
Item	HTML	Image	Audio	Video	Compressed	Formatted	Dynamic	Others
% of Requests	19.47	77.52	0.29	0.06	0.06	0.45	1.91	0.24
% of Bytes	23.25	52.15	3.39	10.07	3.20	5.80	0.82	1.32
Mean Transfer Size	10,060	5,665	98,687	1,322,156	458,526	109,657	3,621	46,141
Median Transfer Size	4,710	2,230	3,641	366,890	46,632	5,728	804	5,052
COV of Transfer Size	5.80	2.98	5.47	2.38	3.80	5.95	23.90	7.49
CANARIE								
Item	HTML	Image	Audio	Video	Compressed	Formatted	Dynamic	Others
% of Requests	20.58	76.38	0.44	0.12	0.10	0.56	1.64	0.19
% of Bytes	17.86	49.01	5.74	12.36	5.12	7.89	0.24	1.78
Mean Transfer Size	10,668	7,890	160,373	1,294,966	656,479	173,987	1,794	115,717
Median Transfer Size	4,982	3,199	10,541	445,901	100,447	6458	1,279	11,148
COV of Transfer Size	6.28	2.30	4.48	2.32	3.15	5.38	2.17	8.93
NLANR								
Item	HTML	Image	Audio	Video	Compressed	Formatted	Dynamic	Others
% of Requests	21.51	68.31	0.26	0.08	0.19	0.62	1.69	0.48
% of Bytes	18.11	39.99	3.32	6.36	7.83	14.58	1.00	8.81
Mean Transfer Size	11,845	8,234	176,737	1,071,632	566,495	330,509	8,330	260,419
Median Transfer Size	4,686	2,435	15,978	438,584	104,275	10,038	1,979	16,244
COV of Transfer Size	26.65	2.98	4.67	2.30	3.17	5.38	24.20	3.30

A substantial portion of the byte transfer volume is accounted for by audio, video, compressed, and formatted document types, which, despite their relatively infrequent access (typically close to 1% of the requests), are large enough to generate 20-30% of the bytes transferred. This observation is substantiated by the large mean and median transfer sizes indicated for these document types, compared to the other document types. In general, the COV values within each document type category are much lower than the COV of transfer sizes for the aggregate data set (see Table 1). The large variation in the mean transfer sizes across the diverse document types helps explain the high COV reported in the aggregate data sets.

3.2 One-Time Referencing

A surprising observation made in previous analyses of Web server workloads [6] was that (regardless of the duration of the access log studied) typically 15-30% of the documents accessed in the log were accessed only once in the log. This so-called "one-time" referencing behavior is of concern for Web caching, since there is clearly no point caching something that will be accessed only once.

The precise cause of this one-time referencing behavior is not fully understood. Several explanations have been proposed. First, it might indicate the vastness of the World-Wide Web, and the low signal-to-noise ratio for much of its content. Second, it might reflect human nature in browsing habits (e.g., once a site has been visited, there is no need to visit it again). Third, it might reflect the behavior of content providers, who might use date-based URL names, or who might redesign or modify Web pages on a regular basis to keep them current, while possibly removing or renaming old pages. Fourth, it may reflect the presence of search engines or Web robots that traverse many pages to construct an index. Fifth, the presence of a high percentage of one-timers might indicate that caching hierarchies are actually working well. Finally, it may be the consequence of document pre-fetching (i.e., prediction) algorithms in some proxies and/or client browsers. All of these hypotheses seem plausible. If any (or all) of them are true, then they indicate a challenging workload environment for Web caching algorithms.

Several other explanations are less plausible. For example, attributing this one-time referencing to the presence of dynamic requests (e.g., CGI) is not possible, since dynamic documents typically account for much less than 5% of the workload in Web server and Web proxy access logs. Attributing one-time referencing to typographical errors made by clients in requested URL names does not make sense, since the access log analyses usually focus on *successful* requests, not errors.

Table 3 summarizes the contribution of one-timers with respect to the number of distinct documents, the total requests, and the total bytes transferred, for the three data sets. One-timers account for a substantial portion of the total requests (18-47%) and total bytes transferred (27-48%). Also, approximately 70% of the documents referenced are one-timers. The latter number is significantly higher than that for Web server workloads [6]. Presumably this is because requests made to a proxy are for a much larger population of Web documents, compared to those made to a single server.

Table 3: One-Time Referencing Behavior in Web Proxy Workloads

Item	USask	CANARIE	NLANR
Distinct Documents	5,510,561	4,571,539	8,482,661
One-Timer Documents	3,753,468	3,470,920	6,056,302
One-Timer/Distinct Documents (%)	68.11	75.92	71.40
Total Requests	21,070,330	7,310,038	24,560,611
One-Timer Requests	3,753,468	3,470,920	6,056,302
One-Timer/Total Requests (%)	17.81	47.48	24.66
Total Bytes Transferred (GB)	177	89	345
Total One-Timer Bytes (GB)	48	43	111
One-Timer/Total Bytes (%)	27.1	48.3	32.2

The predominance of one-time referencing for Web documents highlights the need for novel Web proxy caching policies that can effectively discriminate against one-timers. For example, frequency-based algorithms, such as Least Frequently Used (LFU), tend to perform better than recency-based algorithms, such as Least Recently Used (LRU) [12]. Similar observations have been made for Web server caching algorithms [6, 13, 14].

3.3 Transfer Size Distribution

The next analysis focuses on the transfer size distribution for the documents returned to the requesting clients (either directly by the proxy, or after obtaining the document from a higher-level proxy or the originating server). Of particular interest are the shape of this distribution, the presence of a heavy tail in the distribution, and the impact of the heavy-tailed distribution on Web and network performance.

Figure 1 shows the cumulative distribution function of the transfer size for each proxy server, using a logarithmic scale on the horizontal axis. Almost all the transfers are in the range from 100 to 100,000 bytes, with very few small transfers (say, less than 100 bytes) and very few large transfers (say, more than 100,000 bytes). This distribution is similar to the document size distribution reported for Web clients [7] and for Web servers [6, 11, 15, 16].

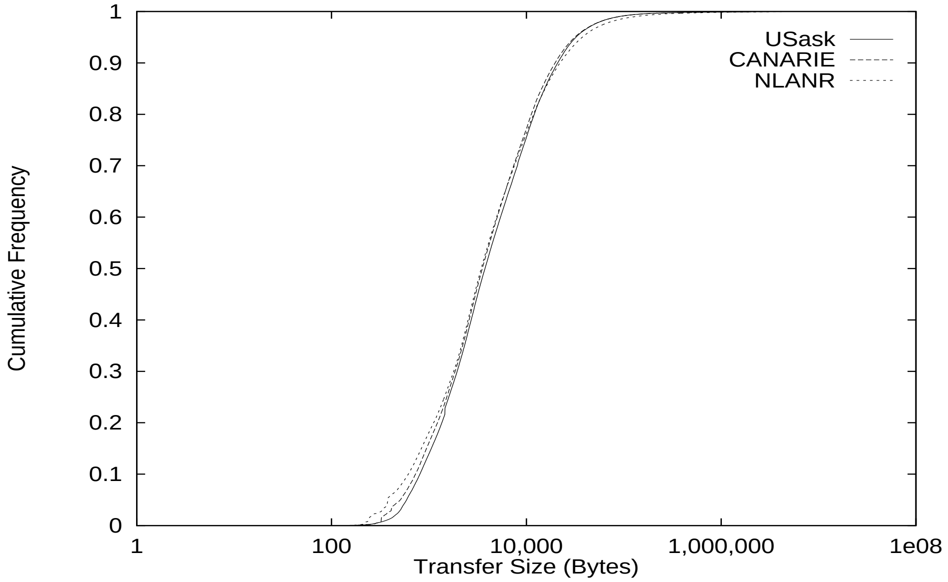

Figure 1: Cumulative Distribution Function for Transfer Sizes, by Proxy

The transfer size distribution in Figure 1 is *heavy-tailed*. In simple terms, a "heavy tail" in a Web document size distribution means that the very large "elephants" (i.e., outliers) in the tail of the distribution, although relatively few in number, are large enough to contribute significantly to the overall traffic volume observed (e.g., skew the mean transfer size distribution, as observed in Table 1).

Figure 2 illustrates the "heavy-tail" transfer size distribution for the USask data set. Figure 2 shows that in the USask data set 75% of all the distinct documents requested are for transfers less than 10,000 bytes in size[5]. About 80% of the references are for documents in this category.

[5] Since multiple references to a document can be associated with different transfer sizes (because of aborted

However, transfers of documents smaller than 10,000 bytes account for only 27% of the total bytes transferred by the proxy to the clients. The tail of the distribution (transfers over 100,000 bytes) accounts for a significant 30% of the total bytes transferred. Similar observations can be made for the other data sets [25]. Therefore, caching smaller documents can increase the hit ratio at the cost of more bytes transferred over the network. To reduce the network traffic volume, proxies can cache larger documents, thus sacrificing the cache hit ratio.

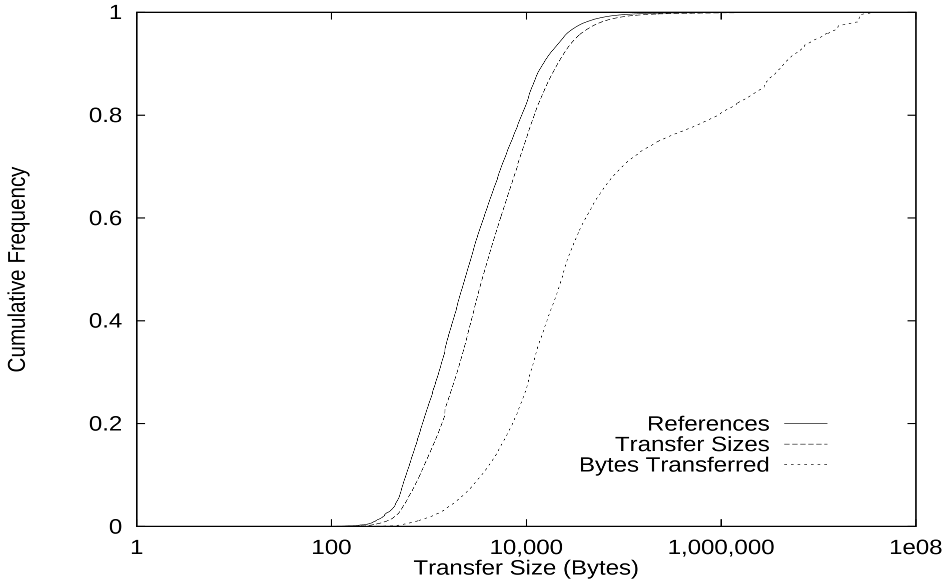

Figure 2: Illustrating *heavy-tail* in the Transfer Size Distribution of the USask data set

The simplest example of a heavy-tailed distribution is the doubly-exponential Pareto distribution; its cumulative distribution function is:

$$F(x) = 1 - (k/x)^{\alpha} \alpha, k > 0, x \geq k$$

The α parameter is known as the tail index [16], and k defines the beginning of the "tail" of the distribution (i.e., it represents the smallest possible value of the random variable in the heavy-tailed distribution). As α decreases, the tail of the distribution becomes heavier [16]. In other words, an arbitrarily large portion of the probability mass may be present in the tail of the distribution as α decreases.

To estimate the tail index α for the transfer size data sets, the approach outlined in [16] and [17] is followed. First, a log-log complementary distribution (LLCD) is plotted for the transfer sizes in the data sets. A LLCD plot graphs $log\bar{\mathrm{F}}(x) = log(1 - F(x))$ versus $logx$, for large x [17]. Heavy-tailed distributions have the following property:

$$\frac{dlog\bar{\mathrm{F}}(x)}{dlogx} = -\alpha, x > k$$

transfers, partial transfers, or document modifications), the average transfer size is considered for all documents referenced more than once.

An estimate of the tail index α is obtained by determining the slope of the LLCD plot for values of x greater than k, using least-squares linear regression [18].

Figure 3 shows the LLCD plot for the NLANR data set, along with the least-squares regression fit for the heavy tail ($k = 10,000$ bytes). Table 4 summarizes the estimated α value for each data set, along with the coefficient of determination (R^2), which assesses the "goodness of fit" for the linear regression. These results show a very strong fit (R^2 is close to 1.0), and α values ranging from 1.07 to 1.27.

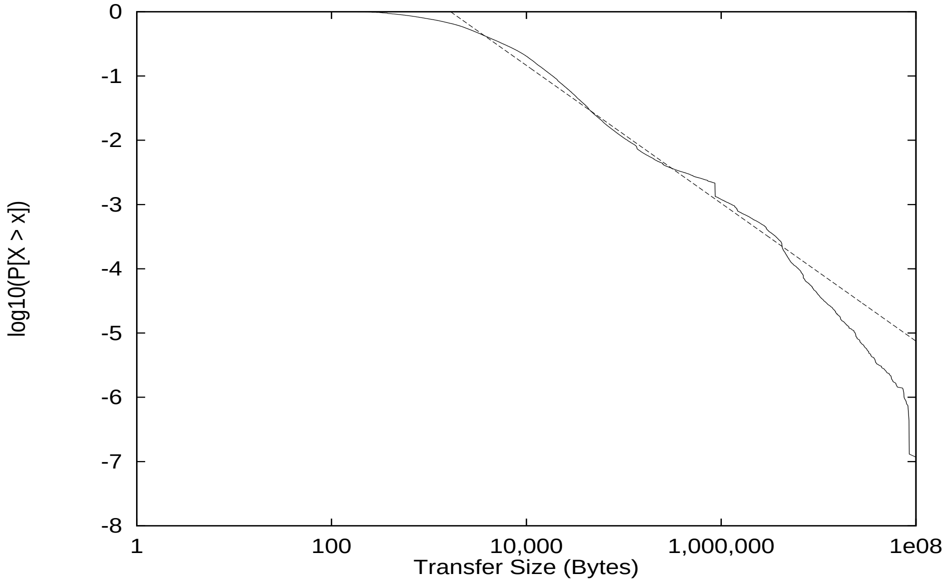

Figure 3: Log-Log Complementary Distribution (LLCD) Plot of Transfer Sizes for the NLANR data set

Table 4: Estimates of α for Heavy-Tailed Transfer Size Distributions

Item	USask	CANARIE	NLANR
α	1.27	1.20	1.07
R^2	0.96	0.97	0.98

It can be concluded that proxy transfer sizes are heavy-tailed, just like server transfer sizes ($0.93 \leq \alpha \leq 1.33$) [6]. Among the three data sets considered, the NLANR data set exhibited the heaviest tail ($\alpha = 1.07$), while the USask data set shows the least heavy tail ($\alpha = 1.27$). One possible explanation for this observation is successful caching of small documents at lower-level proxies, resulting in heavier tails at higher-level caches.

3.4 Document Popularity

Many researchers have noted that the popularity of Web documents is highly uneven [6, 7, 11, 15, 17, 19, 20]. The objective of this analysis is to determine whether the Zipf distribution applies to the proxy document reference streams across different levels of a caching hierarchy.

Zipf's Law [21] states that if items are ranked (r) according to their popularity (P) measured by the frequency of reference for individual items, then the popularity of an item is given by,

$$P \sim \left(\frac{1}{r}\right)^{-\beta}$$

where the exponent β is often close to unity [22].

In the special case where the exponent β is equal to one, the N^{th} most popular item is exactly twice as popular as the $2N^{th}$ item, and so on. The Zipf distribution has been widely used to model many aspects of social and economic behavior [21, 22]. It has also been applied to model memory referencing behavior of computer programs [23, 24], and, most recently, to model Web referencing behavior [20].

To determine whether or not Web proxy document references follow the Zipf distribution, the documents are first ranked, with the most referenced document being assigned a rank of one, followed by the next most referenced document with a rank of two, and so on.

The frequency versus rank plot for the USask data set (on a log-log scale) appear in Figure 4. Visual inspection of the graph suggests that the distribution follows Zipf's law, albeit not as strictly as has been observed for Web servers. Similar observations can be made for the other two data sets [25]. There appears to be some flattening at the most popular end and the least popular end of the plots. This might indicate caching of the frequently requested "hot" documents at browsers and lower-level caches. The middle portion appears to follow the Zipf distribution more accurately. Similar observations are made by Breslau *et al.* [20].

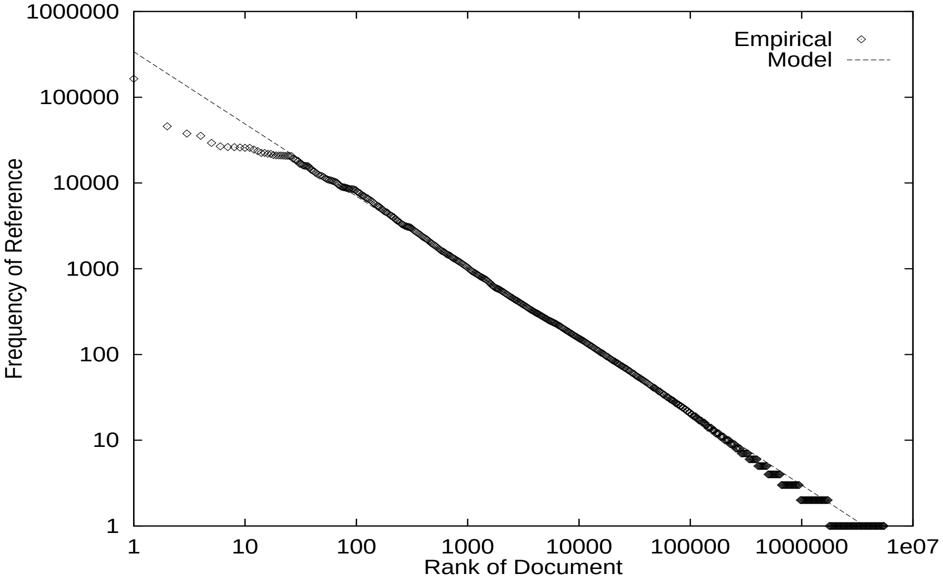

Figure 4: Reference Count versus Rank for the USask data set

To determine the best-fit exponent (β) of the Zipf distribution, the method in [26] is used. This involves grouping all documents that have been referenced an identical number of times and assigning them the same rank. For example, if three documents are referenced 100 times each and are ranked 1000, 1001 and 1002, then the modified data set considers them to be a single data point with a rank of 1001 (i.e., the average of the three ranks) and popularity equal to 100. After obtaining the modified data set, a least-squares fit over the (log-transformed) modified data set is performed. The calculated values for β and the goodness of fit (R^2) values are shown in Table 5. Figure 4 also shows that the Zipf distribution plot obtained with the calculated β value (the dotted line) is very similar to the corresponding frequency versus rank plot. These results suggest that a Zipf-like model can capture the distribution of references seen at a proxy. Note that the exponent β decreases for higher-level proxies. This is most likely due to filtering of popular documents at the browsers and lower-level proxies. Roadknight *et al.* [26] came to similar conclusions after a detailed study of document popularity characteristics.

Table 5: Estimated Slopes for Zipf-Like Referencing Distribution

Item	USask	CANARIE	NLANR
β	0.84	0.77	0.74
R^2	0.96	0.90	0.93

Another way of characterizing the uneven document access patterns is to determine the extent to which references are skewed towards certain documents. A measure of skewness, referred to as *concentration*, was applied to memory and file referencing behavior [23, 27], and later to Web server document accesses [6, 11, 28].

The concentration phenomenon, as applied to documents sorted by their reference counts, is illustrated in Figure 5(a). Non-uniform referencing of Web documents is clearly reflected in this figure, with 30% of the documents accounting for 60-80% of the references. The remaining 70% of the documents (primarily the one-timers) account for the remaining requests.

Concentration of references is also observed when documents are sorted according to the volume of bytes transferred in response to requests for them. Figure 5(b) shows that 10% of the documents account for approximately 90% of the bytes transferred by the proxy to clients.

Higher-level proxies (i.e., CANARIE, NLANR) receive requests from clients that are typically lower-level proxies. Therefore, the concentration of requests at these proxies might be due to the overlap in requests from different clients, as frequent requests from users generally get captured by the browsers and lower-level proxy caches. Among the three data sets, the USask data set shows the most concentration, while the CANARIE data set shows the least. The lower concentration at the CANARIE proxy might be due to its small client population (about 13).

These results suggest that the concentration of references is lower at the Web proxy servers than at the Web servers [6]. This observation makes sense intuitively, since the clients of a Web proxy can effectively access any available document in the Web (i.e. the document set is very large). For Web servers, the requests are restricted to a limited set of documents (i.e., the documents present at the Web server).

However, some recent studies have shown that many static documents are never cached [29, 30]. There are three main reasons for these documents not being cached. First, a significant fraction of the responses received from the Web servers contain no `Last Modified` dates, disabling caching of these requests [30]. Second, many requests in the proxy traces are

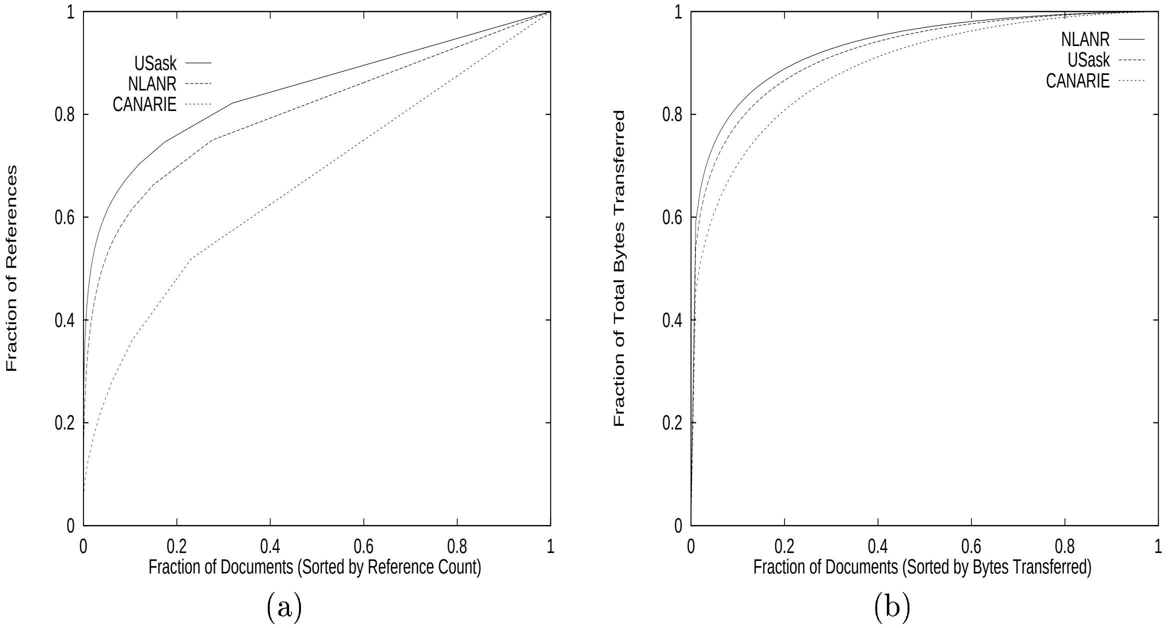

Figure 5: Concentration of References: (a) Documents Sorted by Reference Count; (b) Documents Sorted by Bytes Transferred

associated with "cookies"[6], and most proxies do not cache responses that are cookie-specific. Third, some static documents are associated with an HTTP directive ("no-cache" pragma in HTTP/1.0) that informs the clients that the document should not be cached.

A cursory analysis of the access logs revealed that there are many documents that are never being cached (at the browser, and/or at the proxy). Therefore, part of the concentration of references seen is due to the uncacheability of the documents. Since the access logs used in this analysis did not have the necessary information to distinguish clearly between cacheable and uncacheable documents, no further attempt was made to understand the impact of cacheability of documents.

3.5 Rate of Change of Documents

As already seen, there are multiple references to many Web documents. However, some instances of multiple references may represent accesses to different versions of a document that has been updated. This type of referencing behavior would yield quite different cache behavior than would multiple accesses to an unmodified document. Therefore, it is important to know if there is any correlation between frequency of access and document modifications. Since Web caches do not cache dynamic files, the following analysis focuses on static[7] documents only.

A document is assumed to be modified when the size associated with a reference to the document is different from the size associated with the document when it was last referenced.

[6] A cookie is a small piece of information sent by Web servers to the browsers, and stored at the browsers. This information is used by the Web server the next time the user visits the Web site.

[7] As outlined in Section 3.4, it is not necessary that all static documents are cacheable. However, for simplicity of analysis, all static documents are considered to be cacheable.

There is some error in this assumption, as the access logs report the bytes transferred from the proxy to the clients, which includes HTTP headers. Since document size is approximated by bytes transferred, modifications are reported when header size changes occur, and when clients abort requests. The *change ratio* of a document is defined to be the ratio of the number of modifications seen to the number of references made since the document was first referenced.

Figure 6 shows the average change ratio of documents as a function of the midpoints of reference count intervals. The change ratio for all documents with reference counts in the interval $[x_i, x_{i+1}]$, where $x_{i+1} = 1.1 * x_i$, were averaged to produce each point in the figure.

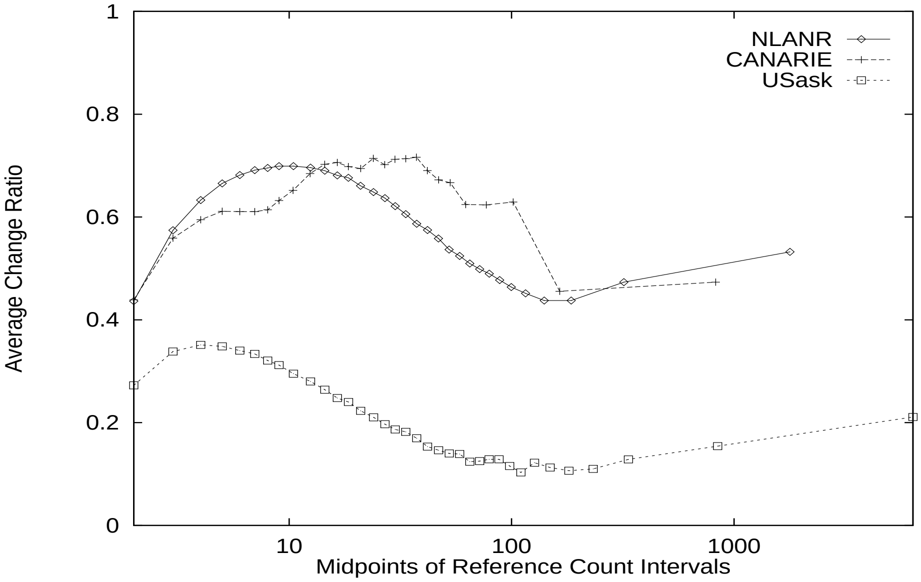

Figure 6: Average Change Ratio as a Function of the Midpoints of Reference Count Intervals

The average change ratio appears to increase initially, then decrease, and finally increase again as the reference count increases. The documents seen in the traces tend to be more frequently modified as one moves higher up in the caching hierarchy because requests from lower-level proxies are forwarded to higher-level proxies when document modifications occur.

4 Conclusions

This paper presented a workload analysis for Web proxy servers. Three different Web proxies were studied: the Web proxy cache at the University of Saskatchewan; the CANARIE Web proxy cache; and an NLANR Web proxy cache at the University of Illinois, Urbana-Champaign. These Web proxies reflect the progression from institutional-level Web proxy cache to a top level Web cache.

The main observations of the study are as follows:

- HTML and Image documents account for 95% of the total requests. Image type documents are consistently the most requested document type, followed by HTML documents.

- Most Web document transfers are small. The mean document transfer size is 7-15 KB, while the median transfer size is 2-3 KB.

- One-timers account for approximately 70% of the documents referenced and between 18-48% of the total requests seen. The percentage of one-timers in Web proxy workloads is higher than that reported for Web server workloads.

- Transfer size distributions are heavy-tailed. The tail of the distribution is heavier at the higher-level proxies than at lower-level proxies.

- The popularity of Web documents does *not* strictly follow Zipf's law as has been reported in Web server workloads, but it does follow a Zipf-like referencing distribution.

- Concentration of references is lower at Web proxy servers that has been reported for Web servers. 30% of the documents account for 60-80% of the references, while 10% of the documents account for about 90% of the bytes transferred. The concentration of references is higher at lower-level Web proxies than at higher-level proxies.

- There appears to be no direct correlation between a document's modification rate and its popularity. The rate of change of documents is higher at higher level proxies.

These observations present a snapshot of Web proxy workload characteristics on one particular Web caching hierarchy at one point in time. Clearly, to comment on the general characteristics at different levels of a caching hierarchy, a study needs to look at more than one caching hierarchy, as was done for Web server [6], proxy [8], and client [7] workloads. Nevertheless, these results and observations can be used to design a flexible synthetic workload generator. Such workload generators can be used to learn more about the impact of different workload parameters on the performance of Web proxy caches, whether hierarchically configured or not.

As for the future, the World-Wide Web is a continually evolving system. Forecasts indicate that by the year 2002, there will be about 320 million Web users, compared to approximately 100 million users in the year 1998 [31]. The number of multimedia services is also increasing, resulting in more audio and video traffic. Therefore, analysis of Web proxy traffic over time needs to be conducted to understand the changes occurring in the traffic characteristics.

References

[1] M. Baentsch, L. Baum, G. Molter, S. Rothkugel and P. Sturm, "World-Wide Web Caching-The Application Level View of the Internet", *IEEE Communications*, Vol. 35, No. 6, June 1997.

[2] S. Glassman, "A Caching Relay for the World-Wide Web", *Computer Networks and IDSN Systems*, Vol. 27, No. 2, pp. 165-174, November 1994.

[3] CERN - European Laboratory for Paricle Physics, CERN httpd Web Server. Available at URL: `http://www.w3.org/Deamon/`

[4] D. Wessels and K. Claffy, "ICP and the Squid Web Cache", *IEEE Journal on Selected Areas in Communication*, Vol. 16, No. 3, pp. 345-357, April 1998.
Available at URL: `http://ircache.nlanr.net/~wessels/Papers/icp-squid.ps.gz`

[5] A. Chankhunthod, P. Danzig, C. Neerdaels, M. Schwartz and K. Worrell, "A Hierarchical Internet Object Cache", *Proceedings of the 1996 USENIX Technical Conference*, San Diego, CA, pp. 153-166, January 1996.
Available at URL: `http://excalibur.usc.edu/cache-html/cache.html`

[6] 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.

[7] C. Cunha, A. Bestavros, and M. Crovella, "Characteristics of WWW Client-Based Traces", Technical Report TR-95-010, Department of Computer Science, Boston University, April 1995.
Available at URL: `ftp://cs-ftp.bu.edu/techreports/95-010-www-client-traces.ps.Z`

[8] G. Abdulla, E. Fox, M. Abrams and S. Williams, "WWW Proxy Traffic Characterization with Application to Caching", Technical Report TR-97-03, Computer Science Department, Virginia Tech., March 1997.
Available at URL: `http://www.cs.vt.edu/~chitra/work.html`

[9] CANARIE, CA*net II sanitized access logs.
Available at URL: `http://ardnoc41.canet2.net/cache/squid/rawlogs/`

[10] National Laboratory for Applied Network Research, NLANR sanitized access logs.
Available at URL: `ftp://ircache.nlanr.net/Traces/`

[11] H. Braun and K. Claffy, "Web Traffic Characterization: An Assessment of the Impact of Caching Documents from NCSA's Web Server", *Computer Networks and ISDN Systems*, Vol. 28, Nos. 1 & 2, pp. 37-51, January 1995.

[12] M. Arlitt, R. Friedrich and T. Jin, "Performance Evaluation of Web Proxy Cache Replacement Policies", Technical Report HPL-98-97, Hewlett Packard Laboratories, May 1998.
Available at URL: `http://www.hpl.hp.com/techreports/98/HPL-98-97.html`

[13] S. Williams, M. Abrams, C. Standridge, G. Abdulla, and E. Fox, "Removal Policies in Network Caches for World-Wide Web Documents", *Proceedings of the 1996 ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Stanford, CA, pp. 293-305, August 1996.

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

[15] P. Barford and M. Crovella, "Generating Representative Web Workloads for Network and Server Performance Evaluation", *Proceedings of the 1998 ACM SIGMETRICS Conference*

on the Measurement and Modeling of Computer Systems, Madison, WI, pp. 151-160, July 1998.

[16] 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. 835-871, December 1997.

[17] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in Web Client Access Patterns: Characteristics and Caching Implications", *World Wide Web Journal*, Vol. 2, pp. 15-28, 1999.
Available at URL: `http://cs-www.bu.edu/faculty/crovella/paper-archieve/traces98.ps`

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

[19] 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)*, Miami Beach, FL, pp. 92-103, December 1996.
Available at URL: `http://www.cs.bu.edu/~best/res/papers/pdis96.ps`

[20] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "On the Implications of Zipf's Law for Web Caching", *Proceedings of IEEE INFOCOM'99*, New York, March 1999.
Available at URL: `http://www.cs.wisc.edu/∼cao/papers/zipf-implications.html`

[21] G. Zipf, *Human Behaviour and the Principle of Least-Effort*, Addison-Wesley, Cambridge, MA, 1949.

[22] M. Kendall, "Natural Law in the Social Sciences", *Journal of the Royal Statistical Society A*, Vol. 124, pp. 1-18, 1961.

[23] R. Bunt and J. Murphy, "The Measurement of Locality and the Behaviour of Programs", *Computer Journal*, Vol. 27, No. 2, pp. 238-245, August 1984.

[24] J. Peachey, R. Bunt and C. Colbourn, "Some Empirical Observations on Program Behaviour with Applications to Program Restructuring", *IEEE Transactions on Software Engineering*, Vol. 11, No. 2, pp. 188-193, February 1985.

[25] A. Mahanti, *Web Proxy Workload Characterisation and Modelling*, M.Sc. Thesis, Department of Computer Science, University of Saskatchewan, September 1999.
Available at URL: `ftp://ftp.cs.usask.ca/pub/discus/thesis-mahanti.ps.Z`

[26] C. Roadknight, I. Marshall and D. Vearer, "File Popularity Characterisation", *Proceedings of the 2nd Workshop on Internet Server Performance (WISP 99)*, Atlanta, Georgia, May 1999.
Available at URL: `http://www.cc.gatech.edu/fac/Ellen.Zegura/wisp99/papers/roadknight.ps`

[27] C. Williamson and R. Bunt, "Characterizing Short Term File Referencing Behaviour", *Proceedings of the Fifth Annual IEEE Phoenix Conference on Computers and Communications (IPCC '86)*, Phoenix, AZ, pp. 651-660, March 1986.

[28] 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.

[29] R. Caceres, F. Douglis, A. Feldmann, G. Glass, and M. Rabinovinch, "Web Proxy Caching: The Devil is in the Details", *Performace Evaluation Review*, Vol. 26, No. 1, pp. 11-15, December 1998.

[30] C. Wills and M. Mikhailov, "Examining the Cacheability of User-Requested Web Resources", *Proceedings of the 4th International Web Caching Workshop*, San Diego, CA, March/April 1999.
Available at URL: `http://www.cs.wpi.edu/~mikhail`

[31] L. Lange, "The Internet", *IEEE Spectrum*, Vol. 36, No. 1, pp. 35-40, January 1999.

Biographies

Anirban Mahanti received a B.E. degree in computer science in 1997 from Birla Institute of Technology, Ranchi, India, and an M.Sc. degree in computer science from the University of Saskatchewan in 1999. He is currently working towards a Ph.D. degree in computer science at the University of Saskatchewan. His research interests are in parallel processing and World-Wide Web performance. Anirban can be contacted at: anm474@cs.usask.ca.

Carey Williamson is a Professor in the Department of Computer Science at the University of Saskatchewan. He received a B.Sc.(Honours) in Computer Science from the University of Saskatchewan in 1985, and a Ph.D. in Computer Science from Stanford University in 1991. His general research interests are in computer networks and computer systems performance evaluation. More specific interests include network traffic measurement, workload characterization, network simulation, and Web performance. Dr. Williamson is a member of IEEE, ACM, and SCS. His email address is: carey@cs.usask.ca

Derek L. Eager received the B.Sc. degree in computer science from the University of Regina in 1979, and the M.Sc. and Ph.D. degrees in computer science from the University of Toronto, in 1981 and 1984, respectively. Currently he is a Professor in the Department of Computer Science at the University of Saskatchewan. His research interests are in the areas of performance evaluation, streaming media delivery, and distributed systems.