Submitted to INFOCOM '97.

An Empirical Model of HTTP Network Traffic

Bruce A. Mah
bmah@CS.Berkeley.EDU
The Tenet Group
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720-1776
Tel: (510) 642-8905
FAX: (510) 642-5775

Abstract

The workload of the global Internet is dominated by the Hypertext Transfer Protocol (HTTP), an application protocol used by World Wide Web clients and servers. Simulation studies of this environment will require a model of the traffic patterns of the World Wide Web, in order to investigate the performance aspects of this increasingly popular application. We have developed an empirical model of network traffic produced by HTTP. Instead of relying on server or client logs, our approach is based on gathering packet traces of HTTP network conversations. Through traffic analysis, we have determined statistics and distributions for higher-level quantities such as the size of HTTP items retrieved, the number of items per "Web page", think time, and user browsing behavior. These quantities form a model can then be used by simulations to mimic World Wide Web network applications in wide-area IP internetworks.

Keywords: World Wide Web, HTTP, traffic model, traffic measurements, workload, Internet.

1 Introduction

Simulations have been a long-used tool for the evaluation of computer networks. In order for simulations to yield useful performance data, however, they require accurate models of the system under study and the expected workload to be placed upon that system. In particular, workloads need to capture the various characteristics of network applications. While a number of synthetic workloads have been constructed for more traditional types of network traffic (such as telnet remote logins and FTP file transfers), the design and deployment of new applications requires the development of new traffic models.

One of these new applications is the World Wide Web, a popular approach to retrieving information in the global Internet. The Web (along with the Hypertext Transfer Protocol HTTP, its associated protocol) has come to strongly influence the current state of Internet network traffic.[1] To simulate various aspects of network performance under

1. Just before the NSFNET backbone was transitioned to a new architecture in April 1995, HTTP was the leading source of network traffic across the backbone network, measured both by number of bytes and number of packets transferred [NSFNET95].

contemporary Internet traffic workloads, it is therefore necessary to be able to describe the network usage of this rapidly-growing application.

We have developed an empirically-derived model of HTTP network traffic, designed to provide a synthetic workload to a simulation of a wide-area IP internetwork. It captures a variety of aspects of World Wide Web network activity. At the lowest level, it describes the sizes of individual Web files; through a set of heuristics, these files are classified into multi-file documents, separated by "user think time". At the highest level, our model describes the browsing behavior of users, at a single Web server and between different Web servers. Our model is based on network packet traces, and uses analysis and heuristics to derive information about files and higher-layer units of information.

Section 2 of this paper provides a brief background on the World Wide Web and HTTP. In Section 3, we describe some measurement methodologies used in prior studies of WWW and other Internet applications, and discuss their applicability to this work; we describe our actual approach to measurement-gathering in Section 4. Section 5 describes the various components of our model. We present our experimental results and apply them to our model in Section 6.

2 Background

The World Wide Web (frequently shortened to *WWW* or *Web*) is a collection of documents and services available to the global Internet. Servers furnish these documents on request to clients (also known as *browsers*). Each document (sometimes referred to as a *page*) may consist of a number of files. For example, a multipart document may consist of text represented using the Hypertext Markup Language (HTML) [Berners-Lee95], along with some number of images to be displayed "inline" with the text.

The Hypertext Transfer Protocol (HTTP) [Berners-Lee96] is a request-response protocol designed to transfer the files making up the parts of Web documents. Each transfer consists of the client requesting a file from the server, then the server replying with the requested file (or an error notification). Both the request and reply contain identification and control information in headers. HTTP uses the services of TCP [Postel81] for reliable transport across the unreliable global Internet. In current versions of HTTP, each TCP connection can be used for at most one HTTP retrieval. Future versions of HTTP, as described in [Fielding96], incorporate the work of [Padmanabhan94] and [Mogul95], which propose the reuse of TCP connections for multiple retrievals between the same client and server.

In this paper, we will occasionally take several liberties with terminology. Strictly speaking, Web documents can be transferred by means other than HTTP. In particular, the File Transfer Protocol FTP [Postel85] is used for some portions of the Web, for example document archives where HTTP cannot be deployed for administrative reasons and FTP servers exist already. Thus, the terms "Web server" and "HTTP server" are not strictly synonymous, though we will frequently use them interchangeably. The usage of the terms "Web browser" and "HTTP client" in this document is similar. Contexts in which differentiation is required should be easily apparent.

3 Prior Work

In this section, we summarize three approaches that have been taken in attempting to characterize Internet applications. Two methods, server logs and client logs, have been used in prior investigations of the World Wide Web. The last approach, traffic traces, has been used for past studies of a number of other Internet applications, such as file transfers and remote logins.

3.1 Server Logs

Most Web servers keep logs of the files they have served, for reasons ranging from operational monitoring to collecting demographic information about users. A workload model can be created by processing the logs of a running Web server. In some sense this approach is the easiest to take, because the machinery for collecting model data already exists and, in fact, the data is very likely being collected anyway. Indeed, for some studies, such as [Mogul95] and [Arlitt96], it is appropriate to use a characterization of a stream of HTTP requests arriving at a Web server.

However, there are two principal drawbacks to this approach. One important disadvantage of using server logs is that they cannot easily capture user access patterns across multiple Web servers. In particular, it may be difficult to make any determination about the locality of references during any given user session. Another shortcoming is that current server logs do not capture any aspects of HTTP overheads, such as protocol headers.

3.2 Client Logs

[Crovella96], [Cunha95], and [Catledge95] relied on data gathered by instrumenting the NCSA Mosaic Web browser [Mosaic95] to log all retrievals made during Web user sessions. The instrumented systems were in public computing laboratories in academic environments. These studies were primarily concerned with investigating various characteristics of Web accesses. However, it would reasonable to expect that one could construct a corresponding model, suitable for generating a synthetic workload.

Unlike server logs, this approach captures user accesses between multiple Web servers quite well. In addition, it allows the characterization of the effects of client-side caching of documents (or parts of documents). However, this technique requires that browsers be able to log their requests, or more likely, the availability of source code for the Web browser so that such logging can be added. Source for newer Web browsers, including the popular Netscape Navigator [Netscape96], is generally not available. In addition, supporting a variety of browsers may be difficult if modifications for logging need to be made to each one.

3.3 Packet Traces

Another method of gathering workload data consists of analyzing packet traces taken from a subnet carrying HTTP traffic, typically an Ethernet or other broadcast-style LAN.[2] From the packet traces and from knowledge about the higher-layer protocols used, traffic analysis can yield a model of the behavior of the original application. This approach has been used in a number of other traffic studies, such as [Cáceres91] and [Paxson91], that predate the Web. [Stevens96] analyzes the packets arriving at an HTTP server and presents some interesting statistics and observations. [Danzig91] describes a library of traffic models for common (circa 1991) Internet applications, which is designed for inclusion in network simulators requiring synthetic workloads. [Paxson94] additionally describes analytic models derived from traffic traces, which have a more compact representation than purely empirical models and can be parameterized to more accurately reflect particular networks.

2. It is possible to use this methodology on a point-to-point link acting as a transit network, but such opportunities are less common.

This approach eliminates the principal disadvantages of the two previous methods mentioned. However, it too introduces drawbacks. Models based on application-level logs can easily record higher-level information such as specific files requested, HTTP message types, and document types. While such information could in principle be gleaned from a packet trace, it would involve considerable effort in reconstructing the contents of each TCP connection. In addition, the effects of client caching of documents are more different to ascertain, since only cache misses generate network traffic.

4 Methodology

We chose to use a packet trace-based approach for our model, principally because it allowed us to capture the behavior of individual users and we would be able to use this methodology with the most popular currently deployed HTTP client (Netscape Navigator). While this approach does lose higher-level information such as the actual files accessed, we felt that such a characterization is not essential to a network workload model.

We used the freely-available tcpdump packet capture utility [Jacobson95] running on a DEC Alpha 3000/300 to record packet headers on a shared 10 Mbps Ethernet in the Computer Science Division at the University of California at Berkeley, during four periods in late 1995. This procedure saved the TCP and IP headers of each packet; a small number of payload bytes were also captured. These data were saved to disk for off-line processing.

The subnet examined is a stub network (no transit traffic), one of a dozen or so in use in the Computer Science Division. There are approximately a hundred hosts on this subnet; the majority of them are desktop UNIX workstations, each principally used by a single user. The user community consists primarily of Computer Science graduate students. While no statistics are available on the relative popularity of different Web clients used in this environment, operational experience suggests that the most prevalent is Netscape Navigator [Netscape96]. There are also several WWW servers on this subnet, associated with various research groups.

Most HTTP servers bind to a well-known TCP port (80).[3] By looking for all TCP packets to or from this well-known port, we captured what we believe is the vast majority of HTTP traffic. Table 1 summarizes the traffic traces we gathered for this study. The first three traces were collected as a part of an effort to examine various types of network traffic (not just HTTP traffic); the packet counts from these traces include only those packets attributable to HTTP. The last traffic trace collected HTTP packets only. From these streams of packets, we extracted those comprising HTTP connections originating from clients on the local network.

Start Time	**End Time**	**Number of HTTP Packets**
Tue Sep 19 16:12:33 1995	Thu Sep 21 07:53:22 1995	186068
Wed Oct 11 09:48:53 1995	Thu Oct 12 14:10:16 1995	458264
Wed Nov 1 11:22:47 1995	Thu Nov 2 10:53:12 1995	369671
Mon Nov 20 11:13:36 1995	Sun Nov 26 05:28:17 1995	676256

TABLE 1. Summary of Traffic Traces.

Although we do not have complete packet loss figures for these traces, we did record a packet loss of approximately 6000 out of 44,000,000 packets during the 1 November 1995 trace (before filtering to isolate HTTP packets). These figures yield a packet loss ratio of only 0.014%. Similar packet capture experiments using this hardware and network have produced loss figures consistent with this trace.

5 Model

Our model of HTTP traffic captures logically and physically meaningful parameters of Web client behavior. The traffic traces described in the preceding section provide us with empirical probability distributions describing various components of this behavior. We use these distributions to determine the characteristics of a synthetic workload. In this section, we present the various components of our model, which are summarized in Table 2.

At the lowest level, our model deals with individual HTTP transactions. Each HTTP transfer consists of a single request-reply pair of messages. In the most common case, the client application sends a request for some data; the

3. In a recent study of the characteristics of HTML documents indexed by the Inktomi “web crawler”, approximately 94% of the documents surveyed were accessed via the standard HTTP port [Woodruff96].

Quantity	Units	Description
request length	bytes	HTTP request length
reply length	bytes	HTTP reply length
document size	files	Number of files per document
think time	seconds	Interval between retrieval of two successive documents
consecutive document retrievals	pages	Number of consecutive documents retrieved from any given server
server selection	server	Relative popularity of any Web server, used to select each succeeding server accessed

TABLE 2. Quantities Modeled.

server in turn replies by supplying that data. These data are transmitted over a single TCP connection [Berners-Lee96]. The first two quantities of our model are therefore the *request length* and *reply length* of HTTP transfers.[4]

At first glance, it may seem more appropriate for a model of network traffic to concern itself instead with the number, size, and interarrival times of TCP segments. However, we note that, in particular, packet interarrival times are governed by the TCP flow control and congestion control algorithms. These algorithms depend in part on the latency and effective bandwidth on the path between the client and server. Since this information cannot be known *a priori*, we conclude that an accurate packet-level network simulation will depend on a simulation of the actual TCP algorithms. This is in fact the approach taken for other types of TCP bulk transfers in the traffic model described in [Danzig91].

Web documents employ a model in which a document can consist of multiple files. Thus, a server and client may need to employ multiple HTTP transactions, each of which requires a distinct TCP connection, to transfer a single document. For example, a document could consist of HTML text [Berners-Lee95], which in turn could specify three images to be displayed "inline" in the body of the document. Such a document would require four TCP connections, each serving one HTTP request and reply. Thus, the next higher level of behavior above individual files is naturally the Web document, characterized in terms of the *number of files* needed to represent a document.

4. In Section 6.5, we will show it is actually appropriate to model the request and reply lengths of the first HTTP transfer on a Web page separately from any remaining retrievals for that page. For simplicity, we have postponed discussion of this distinction.

Between Web page retrievals, the user is generally considering her next action. We readily admit the difficulty of reliably characterizing user behavior, due to its dependency on various human factors beyond the scope of this study. However, we can construct a distribution of *user think time* based on empirical observations.

Assuming that users will tend to access documents from the same server during consecutive page retrievals, it is useful to characterize the locality of reference between different Web pages. We therefore define the *consecutive document retrievals* distribution as the number of consecutive pages that a user will retrieve from a single Web server before moving to a new server (either as a result of following hyperlinks in an existing document, or by selecting a completely unrelated document).[5]

Finally, the *server selection distribution* defines the relative popularity of each Web server, in terms of how likely it is that a particular server will be accessed for a set of consecutive document retrievals.

6 Experimental Results

From our traffic traces and subsequent analysis, we derived the various probability distributions for the different components of our model. The distributions used in our model are consistent with existing Web measurement studies. We have summarized the more interesting facets of these measurements in Table 3.

HTTP request sizes show a bimodal distribution.
HTTP reply sizes have a heavy-tailed distribution, and tend to be larger than request sizes.
A simple heuristic based on timing can be used to group individual files into documents.
The number of files per document tends to be small; 80% of documents required less than four file transfers.
HTTP requests to retrieve the first file of a multi-file Web document tend to be longer than subsequent requests.
Files retrieved as the first file of a multi-file Web document tend to be larger than subsequent files.
The number of consecutive documents retrieved from a given server tends to be small. 80% of visits to a server's document space resulted in fewer than six documents being retrieved.

TABLE 3. Selected Measurement Results.

5. Implicit in this component of the model is the additional assumption that all the components of a Web document tend to come from the same server.

6.1 Anomalies

In some cases we noticed odd trends in our data, which indicated a large number of nearly-identical Web documents transferred at periodic intervals. For example, the 11 October trace showed a number of web page retrievals with interarrival times of about five minutes. There were 291 instances of this document transferred, accounting for approximately 20% of those estimated to be transferred during the whole trace. Upon further investigation, we determined that the documents came from a Web server that displayed real-time still images of the San Francisco, CA skyline. The page used an extension to HTML which caused the client to automatically reload the document at regular intervals, thus updating the picture every five minutes [Netscape96]. As these periodic HTTP retrievals were skewing our data, we removed them from our traces prior to further analysis.[6]

6.2 Request Length

HTTP requests are sent from a client to a server. They typically specify a file to retrieve, although they may also provide information to a computation to be performed on the server. Also contained in each request are some identifying information about the user, the client software, and the request itself.

Since the only user bytes sent from client to server are those contained as a part of the HTTP request, we can measure the request sizes by simply counting the number of bytes in the appropriate direction of each TCP connection. The statistics summarizing the requests in our four traces is shown in Table 4.

	19 Sep 1995	**11 Oct 1995**	**1 Nov 1995**	**20 Nov 1995**
Number	5030	5699	3659	18034
Minimum Size	10	40	40	8
Maximum Size	1825	1786	1333	2404
Mean Size	356	327	325	301
Median Size	231	244	235	244

TABLE 4. Summary of HTTP Request Lengths (in Bytes).

6. While it may be argued that these retrievals should contribute to our traffic model since they actually occur in real life, the nature of this model is such that it cannot accurately capture the correlations between successive document retrievals from such a Web client. A model attempting to characterize such periodic Web traffic should explicitly account for this behavior.

The cumulative distribution functions (CDFs) for the request size distributions are shown in Figure 1. The reply sizes in our traces all exhibited a bimodal distribution, with one large peak occurring around 250 bytes and another, smaller one around 1KB. We believe that the former requests correspond to simple file retrievals, while the latter may contain more complex requests such as those generated by HTML forms. However, there is insufficient information in our existing traces to prove or disprove this hypothesis. (Investigating further would require packet traces containing all or most of the payload bytes from each packet.)

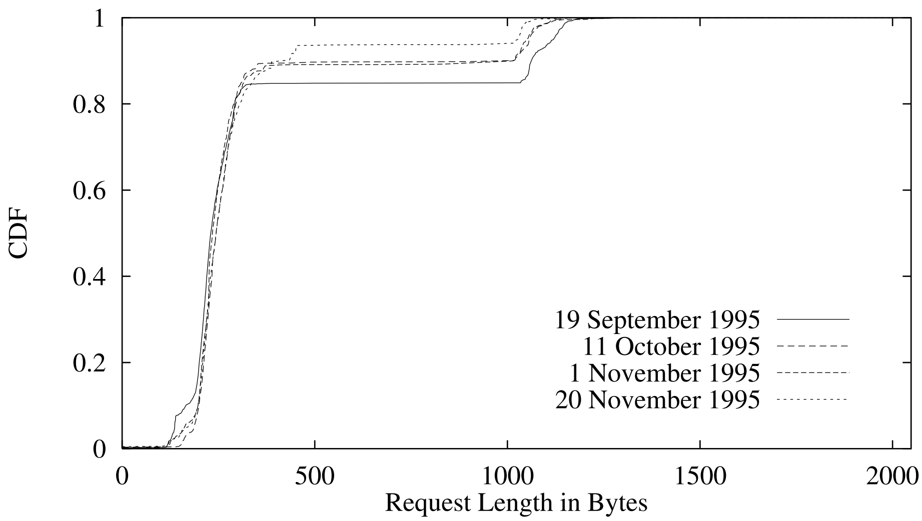

FIGURE 1. Cumulative Distribution Functions of HTTP Request Lengths.

6.3 Reply Length

The HTTP reply consists of the bytes sent from the server to the client. Typically, the reply contains either HTML text or some multimedia data (e.g. an image or audio clip) to be display by the Web client. In the case of an error (e.g. a nonexistent file or access denial), the HTTP reply contains an error message. As with HTTP requests, some identifying information is also included.

In Table 5, we present a summary of the HTTP replies recorded in our four traces. The CDFs for the reply size distributions are shown in Figure 2. We note that two of the maximum file sizes are identical. Upon further investigation, these replies were both generated by downloads of a single large data archive file from a single Web server operated by a local research group.

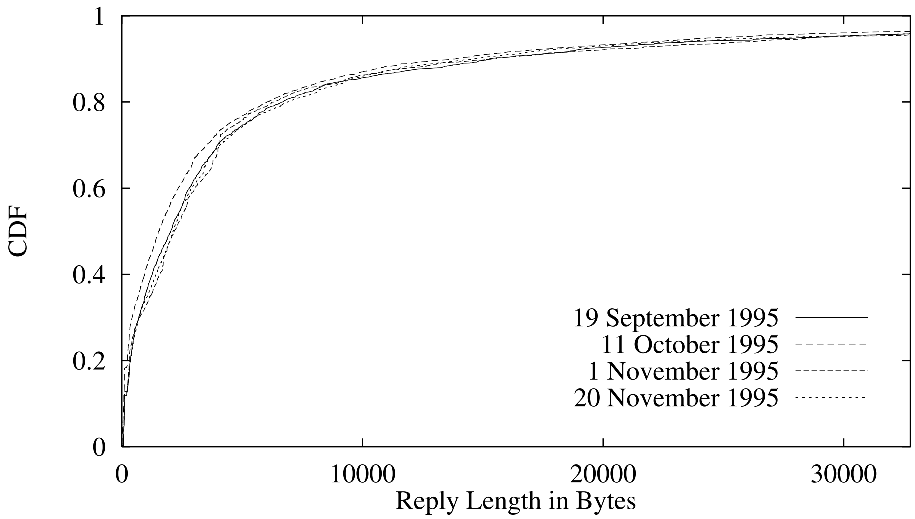

FIGURE 2. Cumulative Distribution Functions of HTTP Reply Lengths.

	19 Sep 1995	**11 Oct 1995**	**1 Nov 1995**	**20 Nov 1995**
Number	5030	5699	3659	18034
Minimum Size	62	81	57	30
Maximum Size	8146976	3270319	1740705	8146976
Mean Size	10664	8899	8319	8812
Median	2035	1532	2179	2127

TABLE 5. Summary of HTTP Reply Lengths (in Bytes).

In each of the traces, the minimum reply length is very short (only tens of bytes). It is likely that these replies represent either errors or "not modified" responses to `If-Modified-Since` (conditional document retrieval) requests. While the actual length of some files may indeed be in the range of tens of bytes, the addition of HTTP headers will make the reply messages somewhat longer.

We note that the maximum reply sizes are rather large (over 1 MB in each of the traces). Further, the means (8–10 KB) are much larger than the median reply sizes (about 2 KB). These characteristics are consistent with distributions of reply sizes that are "heavy-tailed" (with a large amount of the probability mass in the tail of the distribution). It has been in fact demonstrated that WWW file sizes are heavy tailed [Crovella96].

On the assumption that HTTP retrievals generally result in the transfer of a WWW file (and in particular, the assumption that large HTTP replies contain WWW files), it seems natural to expect that HTTP replies would share this characteristic. We repeated the analysis of [Crovella96] on our data, and found that the distributions of reply sizes above 1KB are reasonably well-modeled by Pareto distributions with estimates ranging from $\alpha = 1.04$ to $\alpha = 1.14$.[7] Further details are given in Table 6. By comparison, [Crovella96] arrived at an estimate of $\alpha = 1.06$.

	19 Sep 1995	**11 Oct 1995**	**1 Nov 1995**	**20 Nov 1995**
α	1.05	1.04	1.09	1.14
R^2	0.98	0.99	0.97	0.98

TABLE 6. Estimates of the α Parameter for the Tail of HTTP Reply Size Distributions. R^2 is the coefficient of determination, and takes values in the range $[0 \ldots 1]$. Values near 1 indicate a "good" fit of the regression, and that the simple linear regression used to estimate α can account for nearly all the variation.

6.4 Page Length

Determining the number of files per page is less straightforward. There is no way to determine exactly which TCP connections were transferred as parts of a single document. An HTTP client merely issues the requests for the files making up a given document, in succession.

We therefore use two simple heuristics to determine whether two HTTP connections belong to the same document. First, the two connections must originate from the same IP address, since retrievals from two different client machines cannot possibly belong to the same document. We note that it is possible for two connections from the same IP address to be associated with two unrelated documents, which can happen in the case that two different users begin fetching a document at the same time. However, we evaluate this possibility as remote in our environment, because the end hosts are workstations each used almost exclusively by a single user.

Second, the two connections cannot be separated by "too much time", an interval determined by a parameter we call T_{thresh}. More formally, let c_1 and c_2 be two HTTP connections. Let $S(c)$ be the arrival time of the starting packet

7. The Pareto distribution is a "heavy-tailed" probability distribution with a CDF given by $F(x) = P[X \le x] = 1 - \left(\frac{k}{x}\right)^{\alpha}$, where k is the minimum value of X.

of connection c and let $E(c)$ be the arrival time of the ending packet of connection c. Assuming $S(c_1) < S(c_2)$, then we judge c_1 and c_2 to belong to the same document if and only if $S(c_2) - E(c_1) \le T_{\text{thresh}}$. If $S(c_1) < S(c_2) < E(c_1)$, the two connections overlap and we judge their respective files to belong to the same document. This condition can occur with browsers that use multiple, overlapping TCP connections to improve interactive performance, such as Netscape Navigator. Figure 3 illustrates the role of T_{thresh} in determining the relation between two HTTP connections.

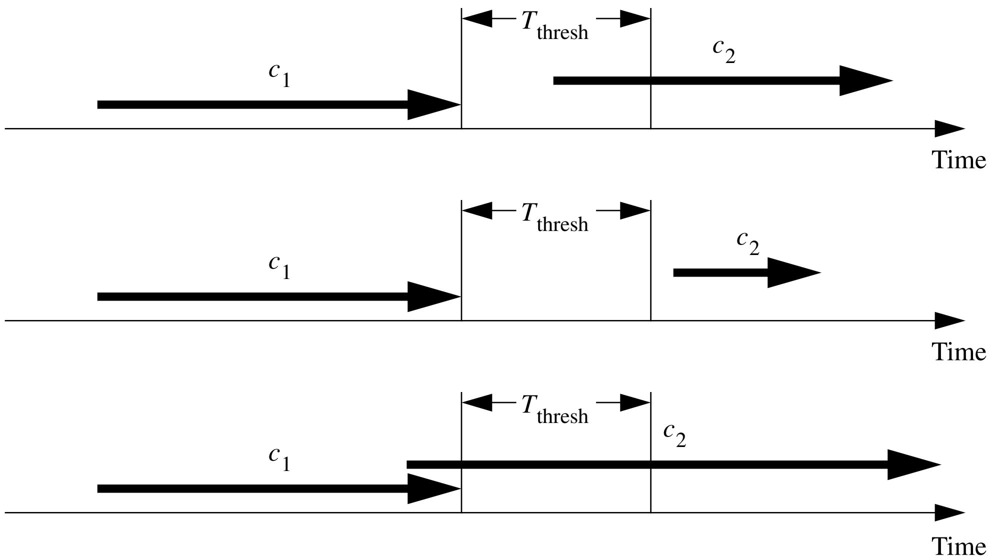

FIGURE 3. Heuristic for Determining the Relation Between Two HTTP Connections. Timelines run left-to-right; TCP connections are represented by thick arrows. In the top timeline, c_2 starts within T_{thresh} time after the end of c_1; thus we judge c_1 and c_2 to belong to the same document. In the center timeline, the gap between c_1 and c_2 is greater than T_{thresh}; thus the two belong to different documents. In the bottom document, c_2 starts before c_1 finishes; in this case the two are judged to belong to the same document.

This heuristic requires, of course, the definition of a suitable value of T_{thresh}. As T_{thresh} becomes very short, it may become smaller than the time necessary for an HTTP client to initiate two retrievals for files belonging to the same page. In this case, connections which really belong to the same Web document will be falsely classified as belonging to different documents. Conversely, as T_{thresh} becomes large, it may become longer than the time for a user to react to the displayed document and select a new document to view. This will make files from different pages appear to be part of the same document.

The analysis in [Crovella96] required a similar classification in order to analyze the distribution of idle times between connections. This analysis classified files separated by less than one second of idle time as belonging to the same document, due to the limitations of the users' reaction time. Idle times greater than 30 seconds were deemed to separate independent documents, as few items would take longer to be processed and displayed. Idle times in the intermediate range were assumed to belong to a "transition" region. According to this reasoning, reasonable values for T_{thresh} can be found in the range $1\,\text{sec} < T_{\text{thresh}} < 30\text{ sec}$.

We picked $T_{\text{thresh}} = 1\text{ sec}$ for this study. The primary influence on our choice of this value is that users will generally take longer than one second to react to the display of a new page and order a new document retrieval. For HTTP clients that perform multiple overlapping file transfers, the time to process and display is not a significant influence on the choice of T_{thresh}, as the various components of a multipart document are downloaded, processed, and displayed in parallel.

Given our choice of an idle threshold, we can characterize the number of files per document, as shown in Table 7. We note that in the survey of HTML documents in [Bray96], slightly more than half of all pages contained either zero or one inlined image, corresponding to either one or two connections per document. Considering that some of our "documents" were actually single-file (thus, single-connection) downloads, which would tend to skew this distribution downward, we feel that our observations are consistent with this statistic, as shown by the cumulative distribution function in Figure 4.

	19 Sep 1995	**11 Oct 1995**	**1 Nov 1995**	**20 Nov 1995**
Mean	2.9	2.8	3.2	3.1
Median	1	1	1	1

TABLE 7. Mean and Median Number of Files Per Document, $T_{\text{thresh}} = 1\text{ sec}$.

We note that although the distribution of the number of files per document does vary as T_{thresh} changes, the distribution is very similar for values around $T_{\text{thresh}} = 1\text{ sec}$. Thus the exact choice of T_{thresh} is not critical to our analysis. Figure 4 illustrates this fact graphically, for the set of HTTP connections recorded in the 19 September 1995 trace.

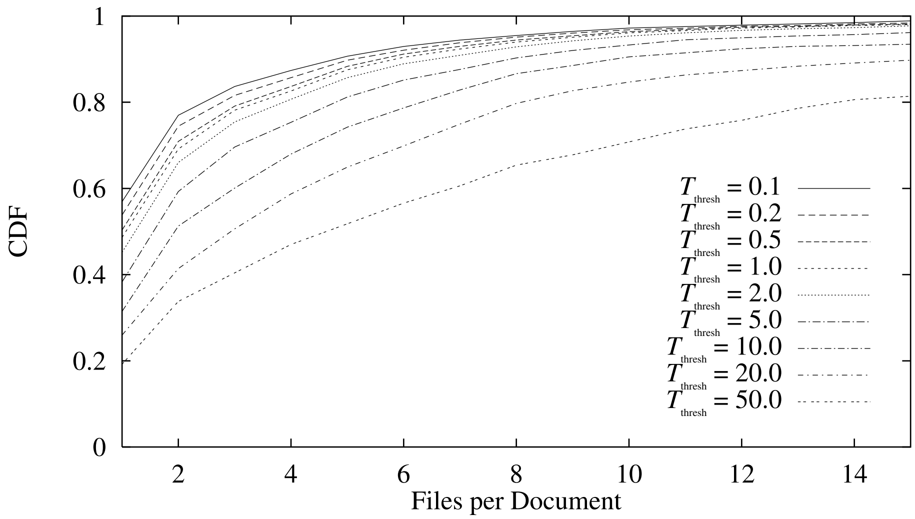

FIGURE 4. Cumulative Distribution Functions of Document Length in Files, 19 September 1995. Curves correspond to varying values of T_{thresh} in seconds.

6.5 Primary and Secondary Retrievals

After classifying files as belonging to different pages, we can partition their retrievals into two classes. The first, which we term *primary retrievals*, consists of the first file of each document. Typically the reply for a primary retrieval will consist of HTML text, but the reply could also consist of an image, a data file to be downloaded, or an HTTP error message.

The other class of retrievals, called *secondary retrievals*, consists of those transactions needed to transfer any remaining files for a document, after that document's primary retrieval. Inlined images (referenced by a primary file consisting of HTML text) are currently the only known example of data transferred by secondary retrievals.

We find that the sizes of requests and replies are slightly different for primary and secondary retrievals. Table 8 summarizes the sizes of primary and secondary requests among our four traces, for $T_{\text{thresh}} = 1\,\text{sec}$. As can be seen, the average and maximum sizes of the primary requests are larger than those of the secondary requests. This tendency is illustrated by the example of Figure 5, which shows the CDF for the primary and secondary request lengths from the 19 September 1995 trace.

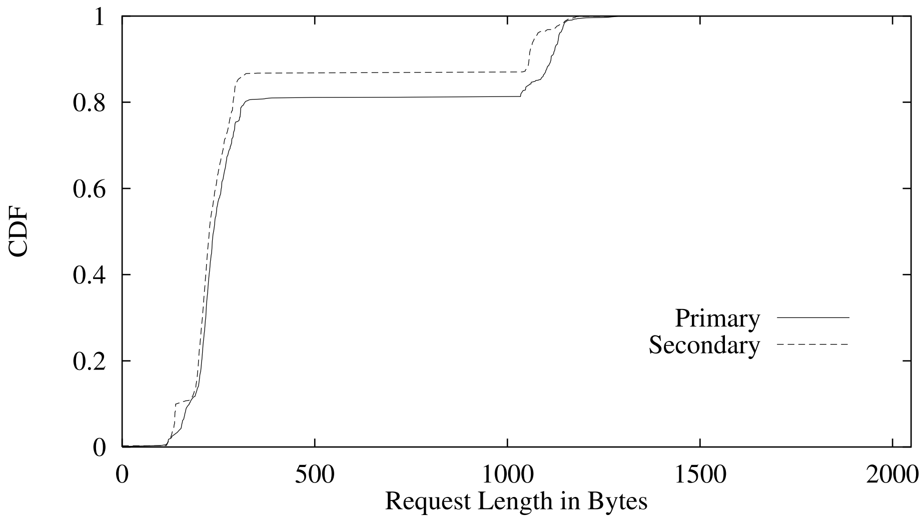

FIGURE 5. Cumulative Distribution Functions of Primary and Secondary Request Lengths, 19 September 1995.

		19 Sep 1995	11 Oct 1995	1 Nov 1995	20 Nov 1995
Primary	Number	1726	2037	1135	5755
	Minimum Size	40	40	40	20
	Maximum Size	1825	1786	1333	2404
	Mean Size	398	346	397	326
	Median Size	235	245	245	243
Secondary	Number	3304	3662	2524	12279
	Minimum Size	106	153	111	100
	Maximum Size	1288	1089	1333	1131
	Mean Size	336	317	292	289.8
	Median Size	227	244	229	245

TABLE 8. Primary and Secondary Request Sizes in Bytes.

In Table 9, we present statistics for the primary and secondary reply sizes, again for $T_{\mathrm{thresh}} = 1\,\mathrm{sec}$. Table 10 shows the results of fitting the reply sizes to Pareto distributions. We find that the tails of primary reply sizes are heaver than those of secondary reply sizes. That is, the estimates for the α parameter of the Pareto distribution are lower for primary replies than for secondary replies, for all our datasets.

To further distinguish the differences between primarily and secondary request sizes, we computed confidence intervals for the estimates of α, and determined that for all four datasets, the corresponding parameters for primary and

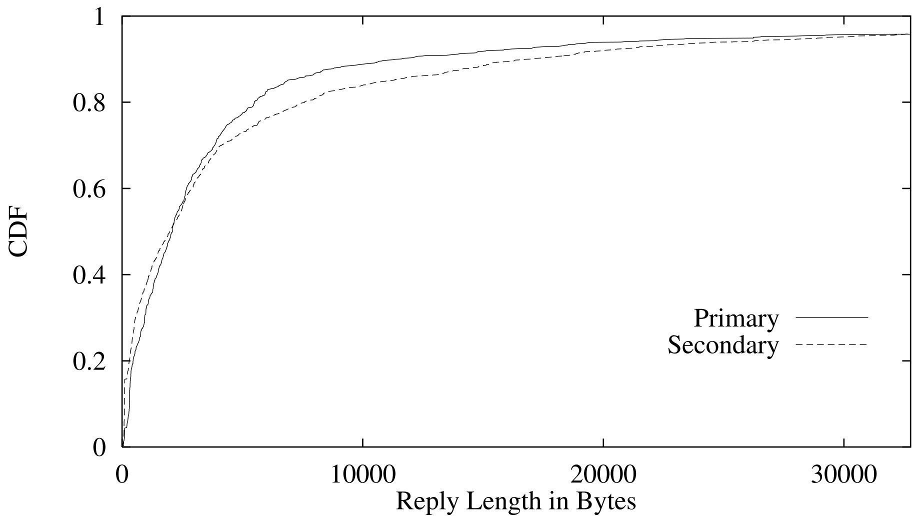

FIGURE 6. Cumulative Distribution Functions of Primary and Secondary Reply Lengths, 19 September 1995.

		19 Sep 1995	11 Oct 1995	1 Nov 1995	20 Nov 1995
Primary	Number	1726	2037	1135	5755
	Minimum Size	106	153	111	100
	Maximum Size	8146976	3270319	1740705	8146976
	Mean Size	17932	14819	12501	14659
	Median Size	2099	2170	2280	2363
Secondary	Number	3304	3662	2524	12279
	Minimum Size	62	81	81	45
	Maximum Size	246212	475554	559970	2413885
	Mean Size	6868	5607	6438	6071.63
	Median Size	1985	1201	2088	1982

TABLE 9. Primary and Secondary Reply Sizes in Bytes.

secondary reply sizes are significantly different at the 90% confidence level. We believe that the differences between the sizes of primary and secondary retrievals are due to the dissimilar types of data being transferred. In particular, we note that arbitrary files downloaded from Web servers are transferred as primary transfers, as are HTML text files. Secondary transfers consist exclusively of inlined images. Based on this analysis, we conclude that it is appropriate to model these two types of retrievals differently.

		19 Sep 1995	11 Oct 1995	1 Nov 1995	20 Nov 1995
Primary	α estimate	0.85	0.88	0.91	0.97
	90% Confidence interval	(0.81, 0.88)	(0.85, 0.91)	(0.84, 0.97)	(0.94, 1.02)
	R^2	0.98	0.99	0.96	0.98
Secondary	α estimate	1.12	1.24	1.21	1.39
	90% Confidence interval	(1.02, 1.23)	(1.14, 1.34)	(1.09, 1.32)	(1.28, 1.51)
	R^2	0.94	0.95	0.94	0.95

TABLE 10. Estimates of α for the Tail of Primary and Secondary Reply Size Distributions.

6.6 User Think Time

Given a selection of T_{thresh}, the empirical distribution of user think times between pages can be characterized by the set of all interconnection idle times T, $T > T_{\text{thresh}}$. In Table 11, we summarize the user think times extracted from the four Web traces. The 20 November 1995 trace had a much longer mean think time than the others. We believe this fact is due to the timing of this particular trace, which covered the American Thanksgiving holiday in late November. The University of California observes this holiday as a four-day weekend, which could conceivably account for some of the long idle times. The CDFs for the user think time distributions for all four distributions is given by Figure 7.

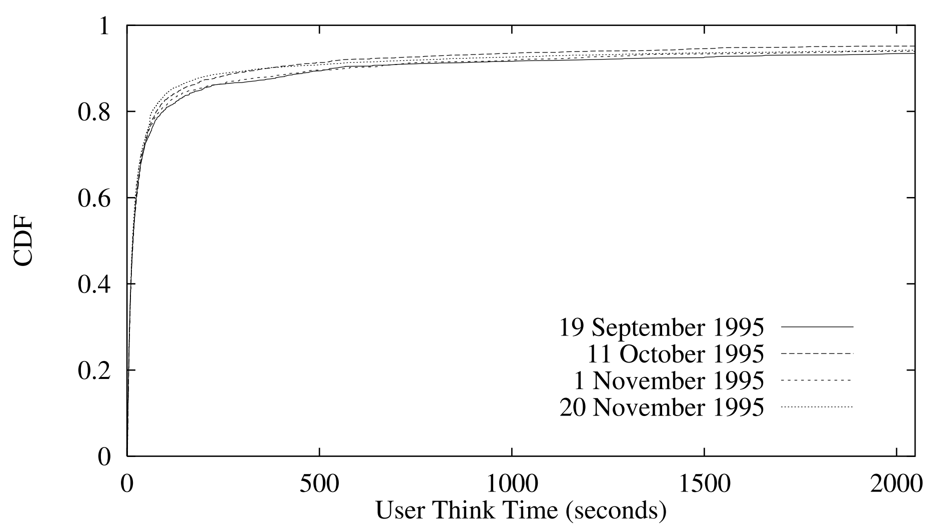

FIGURE 7. Cumulative Distribution Functions of User Think Times.

	19 Sep 1995	11 Oct 1995	1 Nov 1995	20 Nov 1995
Number	1678	1995	1092	5692
Maximum Time	86395	80681	65914	271309
Mean Time	1313	854	837	1915.84
Median Time	15	16	16	14

TABLE 11. User Think Times in Seconds.

6.7 Consecutive Document Retrievals

The current design of many Web document archives is such that users will frequently access documents from the same server in succession. This fact may be important, for example, in network systems that rely on locality of references in allocating virtual circuits or other network resources [Mah95]. Table 12 summarizes the number of consecutive document retrievals from HTTP servers during our network traces. By contrast, [Catledge95] noted that users accessed an average of ten consecutive pages per server, considerably more than the average of four to five document retrievals we observed. We believe that the difference is attributable to the interaction between user browsing strategies and client caching in Web browsers. Users tend to use a browsing strategy that has been described as "spoke and hub", which involves frequent backtracking to already-visited pages. In browsers that implement client side caching, revisited pages will not generate any network traffic (and thus would not appear in a network trace), however they would be counted in a client-side event trace. Thus we would expect our consecutive document retrieval count to be somewhat lower than the corresponding figure in a client access trace by about half, as we observed.

	19 Sep 1995	11 Oct 1995	1 Nov 1995	20 Nov 1995
Number	253	306	171	873
Maximum Documents	37	54	37	112
Mean Documents	4.1	4.3	4.2	4.4
Median Documents	2	2	3	2

TABLE 12. Consecutive Document Retrievals Per Server Access.

In Figure 8, we show the CDF for the consecutive document retrievals distribution from our traces. As can be seen, users tend to switch between servers fairly frequently (the median number of consecutive documents retrieved is approximately two). However, we noted cases in which users continued to access a single Web server for tens of consecutive documents.

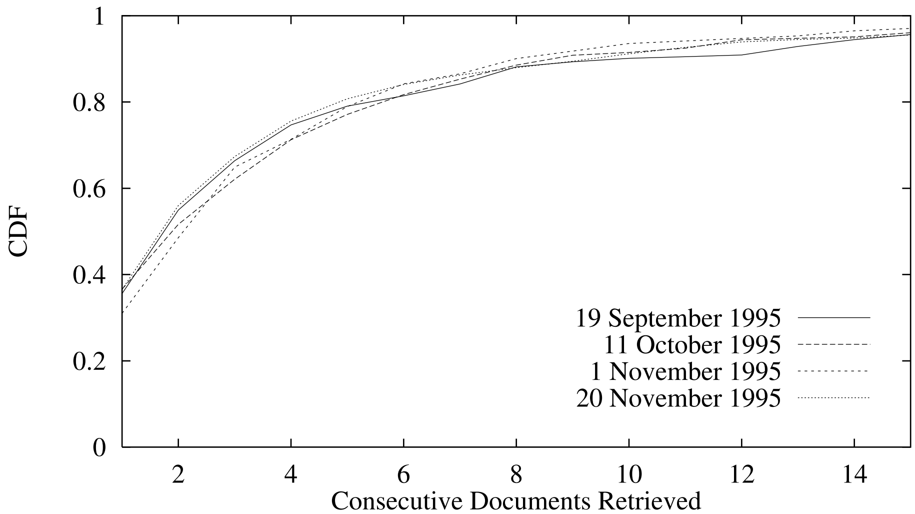

FIGURE 8. Cumulative Distribution Functions for Consecutive Document Retrievals.

6.8 Server Selection

The server selection distribution characterizes the relative popularity of Web servers, from which strings of documents are retrieved. We computed the number of times that any given Web server was used for a set of one or more consecutive document retrievals. In Table 13, we summarize the ten most popular servers for consecutive document retrievals during the 19 September 1995 trace, out of a total of 136 servers accessed as the start of 253 consecutive document retrievals.

The most popular server in this trace (indeed, for all four traces) was the local departmental Web server. Among other items of interest, it also contains homepages for the vast majority of the users of the machines attached to the network being traced. Four of the top ten servers, by this metric, were located on-site.

Given these characteristics, in particular the fact that so many of the servers accessed were local to the tracing site, we believe that we have insufficient information to properly characterize this aspect of our model, based on our existing data. We have chosen instead to approximate the server selection distribution using a Zipf's Law distribution. Zipf's Law is a discrete, heavy-tailed distribution that states that the probability of selecting the ith most popular item in a

Rank	Frequency	Type
1	43	Local
2	11	Local
3	8	Remote
4	7	Remote
5	6	Local
6	6	Remote
7	6	Remote
8	6	Local
9	5	Remote
10	5	Remote

TABLE 13. Top Ten Servers Observed, 19 September 1995. The frequency column shows the number of times any given server was accessed as the start of a stream of consecutive document retrievals. The type of a server reflects whether it is located locally on-site or not.

set is proportional to $1/i$. Originally, it was used to describe the frequency of words in texts, as well as other human-related phenomena [Zipf49]. More recently, this distribution has been applied to the frequency that WWW documents are accessed [Crovella96, Arlitt96]. It would seem reasonable to apply Zipf's Law, or some other heavy-tailed distribution to the access patterns of servers as well, but confirmation of this assertion requires a larger data sample than we have available.

We note here several difficulties in attempting to gauge the popularity of Web servers from IP-layer packet traces. The first problem is that IP-layer packet traces do not reveal the exact hostname originally used to access documents, only the IP address of the server. Hostnames can only be obtained by performing queries to nameservers, which will return the canonical names of hosts, but not their aliases. Thus it may be difficult or impossible to determine that, for example, the machine whose canonical name is `kohler.CS.Berkeley.EDU` is frequently accessed as `http.CS.Berkeley.EDU`.

Another, related problem is that in the case that a hostname maps to multiple IP addresses, it may be difficult to associate accesses to these various IP addresses with a single name. This particular situation may arise in the case of replicated HTTP servers, which rely on randomization in the Domain Name System to spread accesses to a single Web server across multiple machines, as described in [Katz94].

7 Model Representation

When choosing a representation for the various probability distributions making up this traffic model, there are two basic approaches. One is to attempt to fit the observed data to probability distributions that are easily described analytically. A simple analytic representation has the advantage of being compact and (perhaps) easier to use in analysis. This approach was discussed in [Paxson94]; in fact, we performed some rudimentary curve-fitting when analyzing the tails of some of the data samples discussed in Section 6.3 and Section 6.4. However, in circumstances where a data set cannot be described by a well-known distribution (such as the bimodal request size distributions discussed in Section 6.2), this technique cannot easily be used.

The alternative is to represent probability distributions by their CDFs, and to use the inverse transformation method (for example, as described in [Jain91] and applied in [Danzig91]). While requiring more storage and perhaps being slower at generating random values, this approach does have the virtue of being able to represent arbitrary probability distributions.

Due to the flexibility of the latter representation, we chose to maintain the CDF representation for all distributions except a Zipf's Law substitute to the server selection distribution, which is calculated analytically. We based our distributions on the traffic gathered in the 19 September 1995 trace.

A network simulator using our model would need to simulate both the activity of HTTP clients and that of HTTP servers in a network. The behavior of a simple Web browser is illustrated via the pseudo-code in Figure 9; an algorithm for simulating a single-threaded Web server is shown in Figure 10. An earlier version of this model has been implemented and incorporated into the INSANE network simulator, a discrete-event simulator for investigating the performance of IP-over-ATM designs [Mah96]. The simulation of more complex HTTP applications, such as Web browsers that perform multiple, concurrent retrievals, or multi-threaded Web servers, is analogous.

We emphasize that for a meaningful Internet simulation, the actual request and reply data must be regulated by the TCP congestion control and flow control mechanisms, which are not included as a part of this model. It is not suffi-

```
while (notDone) {
    /* select server and number of documents to retrieve */
    /* from that server */
    server = ServerSelection();
    numdocuments = ConsecutiveDocumentRetrievals();

    /* retrieve documents in succession */
    while (numdocuments) {

        /* primary retrieval for document */
        numfiles = DocumentLength();
        requestLength = PrimaryRequest();
        send(requestLength);
        reply = receive();
        numfiles--;

        /* all secondary retrievals for document */
        while (numfiles) {
            requestLength = SecondaryRequest();
            send(requestLength);
            reply = receive();
            numfiles--;
        }

        /* wait for user to think */
        wait(UserThinkTime());
        numdocuments--;
    }
}
```

FIGURE 9. Pseudo-Code for a Simple HTTP Client.

```
while (notDone) {

    /* Get request from client */
    request = receive();

    /* Figure out length of reply based on request type */
    if (request_was_primary) {
        replyLength = PrimaryReply();
    }
    else {
        replyLength = SecondaryReply();
    }

    send(replyLength);
}
```

FIGURE 10. Pseudo-code for a Simple HTTP Server.

cient for network applications to simply transmit data into the network, as such an approach will not accurately model the timing of packets actually transmitted.

8 Conclusions

We have constructed an empirical model of network traffic produced by the Hypertext Transfer Protocol used by World Wide Web applications. This model consists of a number of probability distributions determined by analysis of actual HTTP conversations. From packet traces, we have built up higher layers of communication patterns, from individual HTTP retrievals to Web pages to groups of pages. This approach gives a sufficient level of detail to serve as a component of a workload generator for a packet-level simulation of an IP internetwork being used to carry Web traffic.

Our characterization of WWW-generated network traffic has shown that HTTP requests exhibit a bimodal distribution and that (as revealed in prior studies) sizes of HTTP replies have a heavy-tailed distribution. We have shown that a simple heuristic can be used to separate HTTP transfers into different Web pages, and that the differences between the first and subsequent transfers of a multi-file Web document are statistically significant. We have characterized some aspects of user Web page selection in terms of locality of consecutive documents referenced. Where possible, we have compared the results of our measurements and analysis to other Web measurement studies and found them consistent with those prior results.

9 Future Work

There are of course areas in which this model can be refined; we list several as topics for possible future work. We feel that the Zipf's Law substitute to the server selection distribution could be replaced with an empirical distribution, given an adequately-long trace of network data. It would also be desirable to investigate any correlations between the different components of our model (for example, there may be a correlation between the popularity of a given server and the number of consecutive documents fetched from it). Future protocol developments, such as persistent-connection HTTP, may require new measurement and analysis methodologies. Finally, the conversion of our empirical distributions to closed-form analytic expressions would aid in making the models adaptable to the data and workload found for different types of user communities and document archives.

10 Availability

A subset of the empirical probability distributions gathered in this study, along with C++ classes to easily generate values from these distributions, is available at:

`http://http.cs.berkeley.edu/~bmah/Software/HttpModel`

11 Acknowledgments

The author gratefully acknowledges the comments and suggestions of Kimberly Keeton and Venkata N. Padmanabhan, both of the Computer Science Division at the University of California at Berkeley.

This work was funded by AT&T Bell Labs, the Corporation for National Research Initiatives, (DOE) DE-FG03-92ER-25135, Digital Equipment Corporation, Hitachi, the International Computer Science Institute, Mitsubishi Electric Research Laboratories, (NSF/ARPA) NCR-8919038, and Pacific Bell.

12 References

[Arlitt96] Martin F. Arlitt and Carey L. Williamson. "Web server workload characterization: The search for invariants." In *Proceedings of the ACM SIGMETRICS Conference on Measurement & Modeling of Computer Systems*, pages 126–137, Philadelphia, PA, May 1996.

[Berners-Lee95] Tim Berners-Lee and Daniel W. Connolly. *Hypertext Markup Language – 2.0*. Internet Request for Comments 1886, November 1995.

[Berners-Lee96] Tim Berners-Lee, Roy T. Fielding, and Henrik Frystyk Nielsen. *Hypertext Transfer Protocol – HTTP/1.0*. Internet Draft draft-ietf-http-v10-spec-05, February 1996. This draft is a work in progress which is valid for a maximum of six months from its publication date.

[Bray96] Tim Bray. "Measuring the Web." In *Proceedings of the Fifth International World Wide Web Conference*, Paris, France, May 1996.

[Cáceres91] Ramón Cáceres, Peter B. Danzig, Sugih Jamin, and Danny Mitzel. "Characteristics of wide-area TCP/IP conversations." In *Proceedings of ACM SIGCOMM '91*, Zurich, Switzerland, September 1991.

[Catledge95] Lara D. Catledge and James E. Pitkow. “Characterizing browsing strategies in the World-Wide Web.” In *Proceedings of the Third International World Wide Web Conference*, Darmstadt, Germany, April 1995.

[Crovella96] Mark E. Crovella and Azer Bestavros. “Self-similarity in World Wide Web traffic: Evidence and possible causes.” In *Proceedings of the ACM SIGMETRICS Conference on Measurement & Modeling of Computer Systems*, pages 160–169, Philadelphia, PA, May 1996.

[Cunha95] Carlos R. Cunha, Azer Bestavros, and Mark E. Crovella. “Characteristics of WWW client-based traces.” Technical Report BU-CS-95-010, Computer Science Department, Boston University, July 1995.

[Danzig91] Peter B. Danzig and Sugih Jamin. “tcplib: A library of TCP internetwork traffic characteristics.” Technical Report USC-CS-91-495, Computer Science Department, University of Southern California, Los Angeles, CA, 1991.

[Fielding96] Roy T. Fielding, Jim Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, and Tim Berners-Lee. *Hypertext Transfer Protocol – HTTP/1.1*. Internet Draft draft-ietf-http-v1.1-spec-05, June 1996. This draft is a work in progress which is valid for a maximum of six months from its publication date.

[Jacobson95] Van Jacobson, Craig Leres, and Steven McCanne. “tcpdump software, version 3.0.2,” 1995. This software is available at ftp://ftp.ee.lbl.gov/tcpdump.tar.Z.

[Jain91] Raj Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., New York, NY, 1991.

[Katz94] Eric Dean Katz, Michelle Butler, and Robert McGrath. “A scalable HTTP server: The NCSA prototype.” In *Proceedings of the First International WWW Conference*, Geneva, Switzerland, May 1994.

[Mah95] Bruce A. Mah. “On the use of quality of service in IP over ATM.” Technical Report CSD-95-884, University of California at Berkeley, September 1995.

[Mah96] Bruce A. Mah. *INSANE Users Manual*. Computer Science Division, University of California at Berkeley, May 1996.

[Mogul95] Jeffrey C. Mogul. “The case for persistent-connection HTTP.” In *Proceedings of ACM SIGCOMM ’95*, pages 299–313, Cambridge, MA, August 1995.

[Mosaic95] “NCSA Mosaic software, version 2.6,” 1995. This software is available at http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/.

[Netscape96] “Netscape Navigator software, version 2.02,” 1996. This software is available at http://home.netscape.com/.

[NSFNET95] “NSFNET backbone traffic distribution by service,” April 1995. This document is available at ftp://nic.merit.edu/nsfnet/statistics/1995/nsf-9504-ports.gz.

[Padmanabhan94] Venkata N. Padmanabhan and Jeffrey C. Mogul. “Improving HTTP latency.” In *Proceedings of the Second International World Wide Web Conference*, Chicago, IL, October 1994.

[Paxson91] Vern Paxson. "Measurements of wide area TCP conversations." Masters report, University of California at Berkeley, May 1991.

[Paxson94] Vern Paxson. "Emprically derived analytic models of wide-area TCP connections." *IEEE/ACM Transactions on Networking*, 2(4):316–336, August 1994.

[Postel81] Jon Postel. *Transmission Control Protocol*. Internet Request for Comments 793, September 1981.

[Postel85] Jon Postel and Joyce Reynolds. *File Transfer Protocol (FTP)*. Internet Request for Comments 959, October 1985.

[Stevens96] W. Richard Stevens. *TCP/IP Illustrated, Volume 3*. Addison-Wesley Publishing Company, Reading, MA, 1996.

[Woodruff96] Allison Woodruff, Paul M. Aoki, Eric Brewer, Paul Gauthier, and Lawrence A. Rowe. "An investigation of documents from the World Wide Web." In *Proceedings of the Fifth International World Wide Web Conference*, Paris, France, May 1996.

[Zipf49] George Kingsley Zipf. *Human Behavior and the Principle of Least Effort*. Hafner Publishing Company, New York, NY, 1949.