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

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 IP networks will require a model of the traffic patterns of the World Wide Web, in order to investigate the effects 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 packet traces of HTTP conversations. Through traffic analysis, we have determined statistics and distributions for higher-level quantities such as the size of HTTP files, the number of files per "Web page", and user browsing behavior. These quantities form a model can then be used by simulations to mimic World Wide Web network applications.

1 Introduction

Simulations are a popular tool for the evaluation of computer networks. To yield useful data, however, they require accurate models of the system under study and its expected workload. In particular, workloads need to capture various characteristics of network applications. While a number of synthetic workloads have been constructed for more traditional types of network traffic (such as remote logins and file transfers), new applications require the development of new traffic models.

One such example is the World Wide Web, a popular approach to retrieving information in the global Internet. The Web (along with its associated Hypertext Transfer Protocol) has come to strongly influence the nature of Internet network traffic.[1] To simulate networks under contemporary Internet traffic loads, we therefore require a model of this rapidly-growing application.

Section 2 of this paper provides background on the World Wide Web and HTTP. In Section 3, we describe some measurement methodologies used in prior studies of the Web and other Internet applications; 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. In Section 7, we discuss the representation and use of our model's components.

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, or *page*, may consist of a number of files. For example, a multi-part document could consist of Hypertext Markup Language (HTML) text [3], plus some number of images, with each part in separate files.

The Hypertext Transfer Protocol (HTTP) [4] 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 [26]. In current versions of HTTP, each TCP connection can be used for at most one retrieval. Future versions of HTTP [12] will incorporate a proposal for the reuse of TCP connections for multiple retrievals between the same client and server [18, 23].

3 Prior work

In this section, we summarize three approaches to characterizing Internet applications. Two methods, server logs and client logs, are popular in Web measurements.

1. The last NSFNET traffic measurements in April 1995 showed that HTTP was the leading source of backbone network traffic, measured both by the number of bytes and number of packets transferred [22].

The last approach, traffic traces, has been used for past studies of other Internet applications.

3.1 Server logs

Most Web servers keep logs of the files they have served; this information can be used to create a workload model. Such an approach is fairly easy, because the machinery for collecting model data already exists and, in fact, the data is very likely being collected anyway. Some studies, such as [2] and [18], require such a model of a stream of HTTP requests arriving at a Web server.

However, there are two principal drawbacks to this approach. One large 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 determine the locality of references during any given user session. Another shortcoming is that current server logs do not capture HTTP overheads such as protocol headers.

3.2 Client logs

[7], [8], and [9] relied on data gathered by instrumenting the NCSA Mosaic browser [19] to log all retrievals made during Web user sessions. These studies were concerned with investigating characteristics of Web accesses. Using this data, it would be possible to construct a corresponding model, suitable for generating a synthetic workload.

Unlike server logs, this approach captures user accesses between multiple Web servers. It can also characterize the effects of client document caching. However, this technique requires that browsers be able to log their requests, currently a rare capability.

3.3 Packet traces

A third method of gathering data consists of collecting packet traces taken from a subnet carrying HTTP traffic, typically an Ethernet or other shared-media LAN. From the packet traces and 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 [6] and [24], that predate the Web. [27] analyzes the packets arriving at an HTTP server and presents some interesting statistics and observations. [10] describes a library of traffic models for common (circa 1991) Internet applications, which is designed for inclusion in network simulators. [25] 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.

This approach eliminates the principal disadvantages of the two previous methods mentioned. However, it too introduces drawbacks. A packet trace misses higher-level information such as specific files requested and document types. In addition, the effects of client document caching are more difficult to ascertain, since only cache misses generate detectable network traffic.

4 Methodology

We chose to use a packet trace-based approach, principally because it allowed us to capture the behavior of individual users and we would be able to record the activity of any HTTP client. While this approach loses higher-level information such as filenames, we felt that such data is not essential to a network workload model.

We used the `tcpdump` packet capture utility [13], running on a DEC Alpha 3000/300, to record TCP/IP 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.

The subnet examined is a stub network (no transit traffic) with approximately one hundred hosts; almost all are UNIX workstations used principally 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 in this environment, operational experience suggests that the most prevalent is Netscape Navigator [20]. There are also several Web servers on this subnet, associated with various research groups.

Most HTTP servers bind to TCP port 80.[2] By gathering 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 our traffic traces. The first three traces were collected as a part of an effort to examine various types of LAN 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 for HTTP connections originating from clients on the local subnet. Our traces captured activity from between forty to sixty active client hosts.

Although we do not have complete packet loss figures for these traces, we recorded the loss of approximately 6000 out of 44,000,000 packets during the 1 November

2. 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 [28].

Starting Date	Duration (hr:min)	Packets
19 Sep 1995	39:40	186,068
11 Oct 1995	29:31	458,264
1 Nov 1995	25:30	369,671
20 Nov 1995	138:14	676,256

TABLE 1. Summary of Traffic Traces.

1995 trace (before filtering to isolate HTTP packets). These figures yield a ratio of only 0.014%. Similar packet capture experiments using this hardware and network have produced figures consistent with this loss rate.

5 Model

Our model of HTTP traffic captures logically meaningful parameters of Web client behavior, such as file sizes and "think times". 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 a synthetic workload. In this section, we present the components of our model, which are summarized in Table 2.

Quantity	Description
request length	HTTP request length
reply length	HTTP reply length
document size	Number of files per document
think time	Time between retrieval of two successive documents
consecutive document retrievals	Number of consecutive documents retrieved from any given server
server selection	Used to select each succeeding server accessed

TABLE 2. Quantities Modeled.

At the lowest level, our model deals with individual HTTP transfers, each of which consists of a request-reply pair of messages, sent over a single TCP connection. We model both the *request length* and *reply length* of HTTP transfers.[3]

At first glance, it may seem more appropriate for a model of network traffic to deal with the number, size, and interarrival times of TCP segments. However, we note that these quantities 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*, 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 [10]. In a similar fashion, our model generates transfers which need to be run through TCP's algorithms; it does not generate packet sizes and interarrivals by itself.

A Web document can consist of multiple files. A server and client may need to employ multiple HTTP transactions, each of which requires a separate TCP connection, to transfer a single document. For example, a document could consist of HTML text [3], 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 carrying one request and one reply. The next higher level above individual files is naturally the Web document, which we characterize in terms of the *number of files* needed to represent a document.

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

Assuming that users tend to access strings of documents from the same server, we 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 one.[4]

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 probability distributions for the different components of our model. They are consistent with existing Web measurement studies. We have summarized the more interesting facets of these measurements in Table 3.

3. In Section 6.5, we show it is appropriate to model the first HTTP transfer on a Web page separately from subsequent retrievals for that page. For simplicity, we have postponed discussion of this distinction.

4. We assume that all the components of a Web document come from the same server.

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 page tend to be longer than subsequent requests.
The first file of a Web page tends 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.

6.1 Anomalies

In some cases we noticed odd trends in our data, which indicated a large number of nearly-identical Web documents, transferred periodically. For example, the 11 October 1995 trace showed a number of retrievals with interarrival times of about five minutes. There were 291 instances of these transfers, accounting for approximately 20% of those transferred during the whole trace. Upon further investigation, we found that they contained real-time, still images of San Francisco. They used an extension to HTML which caused clients to automatically reload them every five minutes, thus updating the picture. As these (and other) periodic HTTP retrievals were skewing our data, we removed them from our traces.[5]

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 on the server.

The only data sent from client to server consists of the HTTP request, so we measured the request sizes by simply counting the number of bytes in the appropriate direction of each TCP connection, summed over all packets. We observed mean request lengths of about 320 bytes, with medians around 240 bytes.

5. While it may be argued that these retrievals should contribute to our traffic model since they actually occur in real life, this model cannot accurately capture the correlations between successive, periodic document retrievals. 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 request sizes in our traces all exhibited a bimodal distribution, with one large peak occurring around 250 bytes and another, smaller one around 1 KB. 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 confirm this hypothesis. (Investigating further would require packet traces containing the payload bytes from each packet.)

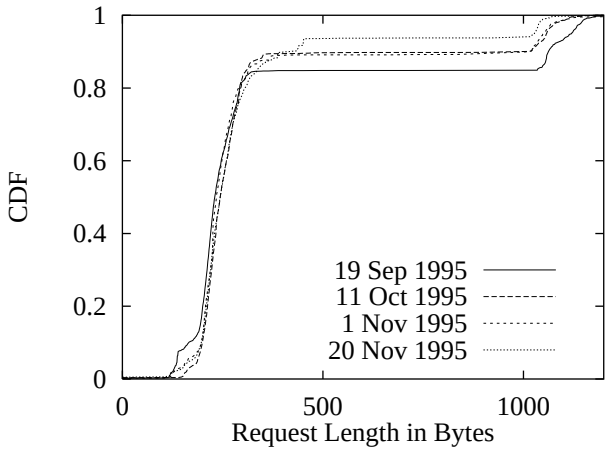

FIGURE 1. HTTP Request Lengths.

6.3 Reply length

Each HTTP reply consists of the bytes sent from server to client. Typically, the reply contains either HTML text or some multimedia data (e.g. an image or audio clip) to be displayed by the Web client. In the case of an error, the HTTP reply contains an error message.

The CDFs for the reply size distributions are shown in Figure 2. The mean file sizes ranged from 8–10 KB, with median files around 1.5–2.0 KB. We note that although many replies were short, we recorded some as long as 8 MB (upon further investigation we determined that they were binary file transfers).

In each of the traces, the minimum reply length was very short (tens of bytes). These replies probably represent either errors or "not modified" responses to conditional document retrieval requests. While some files may indeed be this short, the addition of HTTP headers lengthens the reply messages.

We note that the maximum reply sizes were rather large (over 1 MB in each of the traces). Further, the means (8–10 KB) were 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

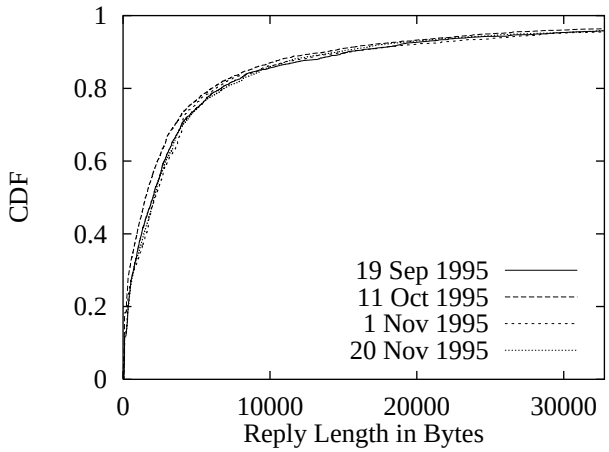

FIGURE 2. HTTP Reply Lengths.

large amount of the probability mass in the tail of the distribution), as shown for WWW file sizes in [8].

Assuming 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 [8] on our data, and found that reply sizes above 1 KB are reasonably well-modeled by Pareto distributions with α estimates ranging from $\alpha = 1.04$ to $\alpha = 1.14$.[6] By comparison, [8] arrived at an estimate of $\alpha = 1.06$.

6.4 Page length

Determining the number of files per page is less straightforward, as we cannot exactly determine the set of TCP connections carrying parts of a single document. We therefore use two simple heuristics to determine whether two HTTP connections belong to the same document. First, they must originate from the same client machine. We note that it is possible for two connections from the same IP address to be associated with two unrelated documents, which can happen if two users fetch a document at the same time. Because our end hosts were workstations used by single users, we feel this occurrence is unlikely.

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 of connection c and let $E(c)$ be the arrival time of the ending packet of connection c. Assuming $S(c_1) < S(c_2)$, we judge c_1 and c_2 to belong to the same document 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 [20].

Figure 3 illustrates the role of T_{thresh} in determining the relation between two HTTP connections c_1 and c_2, between the same client and server. In the top timeline, c_2 starts within T_{thresh} time after the end of c_1; we judge them to belong to the same document. In the center timeline, the inter-file gap is too long, so the files belong to different documents. In the bottom timeline, c_2 starts before c_1 finishes; they must be part of the same document.

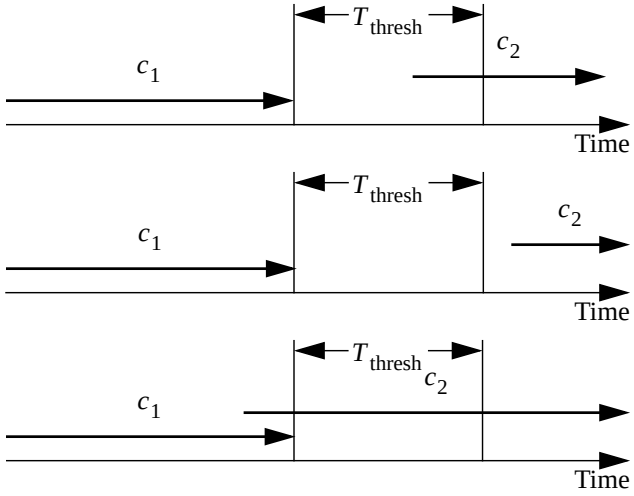

FIGURE 3. Heuristic for Determining Relation Between Two HTTP Connections.

This heuristic requires 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 a file retrieval. In this case, related connections will be falsely classified as belonging to different documents. Conversely, as T_{thresh} becomes long, it may become longer than the time for a user to react to the displayed document and request a new one. Files from different pages could then appear to be part of the same document.

The analysis in [8] 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 assumed to delimit documents, as few items would take longer to be processed and displayed. Values in the intermediate range constituted a "transition" region. According to this reasoning, reasonable values for T_{thresh} can be found in the range $1 \text{ sec} < T_{\text{thresh}} < 30 \text{ sec}$.

6. The "heavy-tailed" Pareto distribution has 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 and α is a shape parameter.

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

Given our choice of an idle threshold, we can characterize the number of files per document. The mean number of files per document ranged from 2.8 to 3.2, with the median documents requiring only one file. We note that in the survey of HTML documents in [5], slightly more than half of all pages contained either zero or one inlined image (one or two connections). Considering that some of our "documents" were actually single-file downloads, which would tend to skew this distribution downward, we feel that our observations are consistent with this statistic.

[11] and [21] both analyzed a 5-minute trace from an Internet backbone for characteristics of flows (aggregations of TCP connections). The results in [21] implied average two-way flows of about 20 KB, considerably shorter than our averages of 26–32 KB. [11] found 90% of request flows to be shorter than 10 KB, with the 90th percentile of reply flows less than 100 KB. Our analogous measures (based on the sums of requests and replies for a page) were 7.4–9.0 KB and 168–249 KB respectively. We feel that the discrepancies are due both to the different metrics being examined and the short length of the trace these studies used.

We note that although the number of files per document varies as T_{thresh} changes, the distributions were similar for values around $T_{\text{thresh}} = 1$ sec, as shown in Figure 4. Thus the exact choice of T_{thresh} is not critical to our analysis.

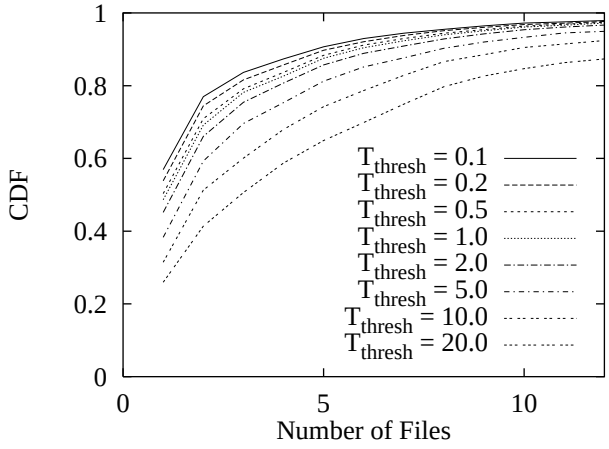

FIGURE 4. Document Lengths, 19 September 1995. Curves correspond to varying values of T_{thresh} in seconds.

6.5 Primary and secondary retrievals

After classifying files into 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 consists of HTML text, but the reply could also be an image, a data file, or an HTTP error message.

The other class of retrievals, called *secondary retrievals*, consists of any remaining files for a document, after its primary retrieval. Inlined images (referenced by a primary file consisting of HTML text) are the only known secondary retrievals.

We found that the sizes of requests and replies are slightly different for primary and secondary retrievals. Primary requests tend to be larger than secondary requests (for $T_{\text{thresh}} = 1$ sec, the median primary and secondary sizes were approximately 240 and 230 bytes respectively). This tendency is illustrated by the example of Figure 5.

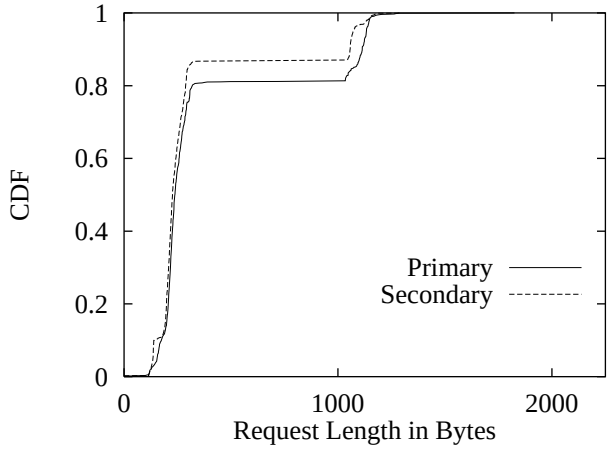

FIGURE 5. Primary and Secondary Request Lengths, 19 September 1995.

Primary reply sizes were also larger. The median primary replies ranged from 2.0–2.4 KB, while the median secondary replies were between 1.2–2.0 KB. When we fitted the reply sizes to Pareto distributions, the α parameter estimates were lower for primary replies than for secondary replies ($\alpha \in [0.85, 0.97]$ vs. $\alpha \in [1.12, 1.39]$).

To further distinguish the differences between primary 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 secondary reply sizes are significantly different with 90% confidence. 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

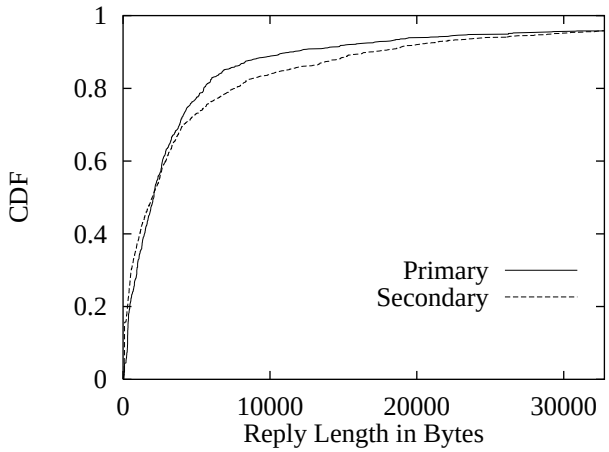

FIGURE 6. Primary and Secondary Reply Lengths, 19 September 1995.

of inlined images (many of which are small bitmaps). Based on this analysis, we conclude that it is appropriate to model these two types of retrievals differently.

6.6 User think time

Given a selection of T_{thresh}, the empirical distribution of user think times between pages is the set of interconnection idle times , $T > T_{thresh}$. The median think times were short, only about 15 seconds. The 20 November 1995 trace had a much longer *mean* think time than the others (1900 vs. about 1000 seconds). We note that this trace spanned the American Thanksgiving holiday in late November, which could explain the long idle times. The CDFs for user think times are given by Figure 7.

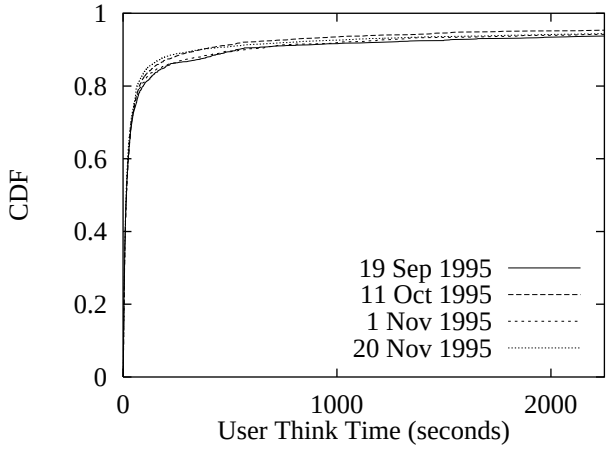

FIGURE 7. Cumulative Distribution Functions of User Think Times.

6.7 Consecutive document retrievals

The design of many Web sites is such that users will frequently access documents from the same server in succession. This fact may be important, for example, in networks that rely on locality of references in allocating virtual circuits or other network resources [17]. In our traces, users downloaded about four pages per server on average, with a median visit of two documents. By contrast, [7] noted that users accessed an average of ten consecutive pages per server. 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), but they would be counted in a client-side trace. Thus, we would expect our consecutive document retrieval count to be somewhat lower than that for a client access trace by about half, as we observed.

In Figure 8, we show the CDF for the consecutive document retrievals distribution. As can be seen, users tend to switch between servers fairly frequently. However, we noted cases in which visits to a server lasted for tens of consecutive documents.

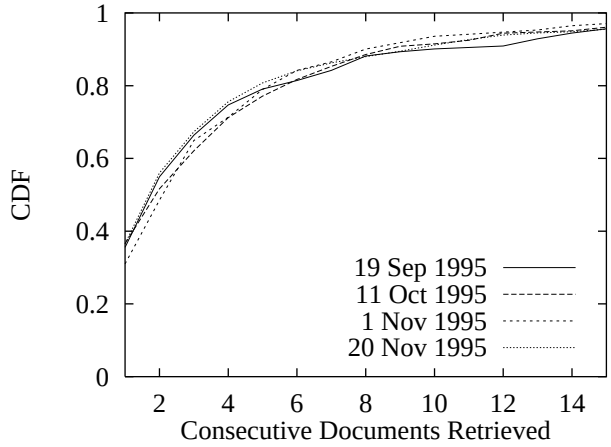

FIGURE 8. Consecutive Document Retrievals.

6.8 Server selection

The server selection distribution characterizes the relative frequency that each Web server was visited for some set of consecutive document retrievals. Using this metric, the most-visited server in this trace (indeed, for all four traces) was the local departmental Web server. It contains homepages for the vast majority of the users, as well as the

initial document for many users. Four of the top ten servers, by this metric, were located on-site.

Given 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. 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 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 [29]. More recently, it has been applied to the access frequency of Web documents [2, 8]. It seems 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 had available.

In general, gauging the identity of Web servers from IP-layer packet traces is difficult. First, it is impossible to determine which of a machine's aliases were used to access it (given only its IP address). Second, if a hostname maps to multiple IP addresses (as is the case for the replicated Web server described in [15]), it may be difficult to associate these addresses with a single name.

7 Model representation

There were two possible representations for this traffic model. One approach was 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) easy to manipulate. This approach was used in [25]; in fact, we performed some rudimentary curve-fitting in Sections 6.3 and 6.4. However, if a data set cannot be described by a well-known distribution (such as the bimodal request size distributions in Section 6.2), this technique cannot easily be used.

The alternative was to represent probability distributions by their CDFs, and to use the inverse transformation method (for example, as described in [14] and applied in [10]). While requiring more storage and perhaps being slower, this approach can represent arbitrary distributions.

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

An earlier version of this model was incorporated into the INSANE network simulator, designed to help investigate the performance of IP-over-ATM designs [16]. It uses the distributions from this model to mimic the patterns of requests and replies sent by HTTP clients and servers.

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 sufficient for network applications to simply transmit data into the network, as such an approach will not accurately model the timing and sizes 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-layer communication patterns, from individual HTTP retrievals to Web pages to groups of pages. This approach yields a sufficient level of detail to serve as a component of a workload generator for a packet-level simulation of an IP internetwork.

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 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

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, between the popularity of a given server and the number of consecutive documents fetched from it). The constantly-changing nature of Web traffic calls for updates to this model to track trends over time. Persistent-connection HTTP will require new measurement and analysis methodologies.[7] Finally, the conversion of our empirical

7. We note that the advanced deployment of persistent-connection HTTP forces new analysis techniques even for late-1996 traffic. Examples can be found in recent versions of Netscape Navigator [20] and the Apache Web server [1].

distributions to closed-form analytic expressions would help make the models applicable to more environments.

10 Availability

A subset of the probability distributions gathered in this study, along with sample code, is available at:

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

11 Acknowledgments

The author gratefully acknowledges the comments of Kimberly Keeton and Venkata N. Padmanabhan, as well as the anonymous INFOCOM reviewers.

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

[1] The Apache Group. *Apache software*, 1996. This software is available at http://www.apache.org.

[2] 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.

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

[4] Tim Berners-Lee, Roy T. Fielding, and Henrik Frystyk Nielsen. *Hypertext Transfer Protocol – HTTP/1.0*. Internet Request for Comments 1945, May 1996.

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] Rusty Eddy. HTTP analysis of IP level traces, February 1996. This document is available at http://catarina.usc.edu/eddy/http-traffic/http-traces.html.

[12] 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-07, August 1996.

[13] 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.

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

[15] 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.

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

[17] Bruce A. Mah. *Quality of Service and Asynchronous Transfer Mode in IP Internetworks*. PhD dissertation, University of California at Berkeley, December 1996.

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

[19] National Center for Supercomputing Applications. *NCSA Mosaic software*, July 1995. This software is available at http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/.

[20] Netscape Communications Corporation. *Netscape Navigator software*, 1996. This software is available at http://home.netscape.com/.

[21] Peter Newman, Tom Lyon, and Greg Minshall. Flow labelled IP: A connectionless approach to ATM. In *Proceedings of IEEE INFOCOM '96*, pages 1251–1260, San Francisco, CA, March 1996.

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

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

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

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

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

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

[28] 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.

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