Statistical Characterization of a World Wide Web Browsing Session

Sunil U. Khaunte
School of Electrical Engineering

John O. Limb
College of Computing

Georgia Institute of Technology
Atlanta, GA 30332-0280
E-mail: (sunil,limb)@cc.gatech.edu

Abstract

The growing availability of Internet access has led to a sharp increase in the use of the World Wide Web(WWW). In the near future, a significant component of the Internet traffic into and out of the home is expected to be WWW related traffic. If we are to design broadband access systems to carry this traffic, it is important to be able to model it accurately, with the hope that we can track and predict the impact of this relatively new type of traffic. The upstream packets presented by a subscriber engaged in a web browsing session, cannot be modelled as an independant one-way source since the actual inter-arrival between the packets depends on the feedback mechanisms and restrictions imposed by the transport/application layer protocols. We need to model the complete packet exchange between the WWW Client/Server to generate realistic source traffic for such systems. In this paper we report statistics on different parameters of the WWW browsing session obtained from actual WWW traces. We also present empirically derived analytical models for the different parameters . The statistical characterization of the different parameters directly lends itself to the development of a more realistic WWW-browser packet generator the basic details of which are addressed in this paper. This WWW traffic generator will be very useful in testing the performance of media access protocols and resource reservation algorithms in broadband access networks.

1 Introduction

The World Wide Web(WWW) is a distributed information retrieval system overlaying the Internet. Using a WWW client(browser), any user connected to the Internet, can retrieve vast amounts of information, stored on different remote computers all around the world. Internet traffic statistics[8], indicate that WWW traffic accounts for more than 25% of the Internet traffic, and is currently growing more rapidly than any other Internet traffic type. In the early days of the Web, most communication was between researchers. Nowadays, the web browsers are getting increasingly user friendly and the large variety of information available on the Web is of interest to researchers and the general public alike. There is a tremendous scope for home based shopping/business opportunities on the Web. Considering these points, one could conjecture that with the future deployment of residential broadband access networks, a significant amount of the subscriber traffic leaving the home will be WWW related traffic.

Most cable-based residential broadband access networks, use a contention based reservation scheme for the upstream channel. Each active station on the network needs to contend for the upstream channel in order to make a reservation request [14]. The actual packets are then sent on the upstream channel when the request has been granted by the bandwidth controller. For such schemes, it is very useful to model the packet flow out of the subscriber, as it provides an insight into the load presented by an individual subscriber on the upstream channel.

In such cases, the WWW client can be regarded as a bursty traffic source, which generates a series of IP packets during the web page retrieval, for document request and control information, followed by a period of inactivity while the user is viewing the last retrieved web page. The upstream packets generated by a subscriber engaged in a web browsing session cannot be modeled using traditional one-way bursty sources. The generation of new packets by such a client source is not totally independent, but is influenced by the downstream feedback data received in response to the previous packets. Assuming a continuous fill up of the modem queues by a one-way WWW client source, can significantly underestimate the load on the contention channel if piggy-backing schemes are used. In the actual system, piggy-backing of new reservation requests with the transmission of earlier packets will not be possible. This is because the new packet is usually generated after the earlier packet has been cleared through the access network to the remote server, and server responds with some feedback data. Thus a modem needs to make an independent upstream reservation request for most of the packet arrivals using the contention channel.
Our particular focus in this paper is understanding the packet level interaction between a WWW client and a remote Web server. To this end, we have developed a WWW client traffic analysis tool, which uses trace data from the *tcpdump[4]* program, and extracts information on different parameters of the web browsing session. A significant amount of work done in the past has concentrated on modeling aggregate Web Traffic [5,6,7,10,11,12]. Related work is being carried out by Mah[13] at the University of California at Berkeley.

The paper begins with an overview of a typical web page retrieval process in section 2. Sec-

tion 3 provides a brief summary of the WWW traffic modeling work done in the past. In section 4 we describe the WWW data collection and processing set-up, followed by a presentation of our findings in section 5 and discussion on a WWW traffic generator that may be developed based on our findings in section 6.

2 Overview of the web page retrieval process

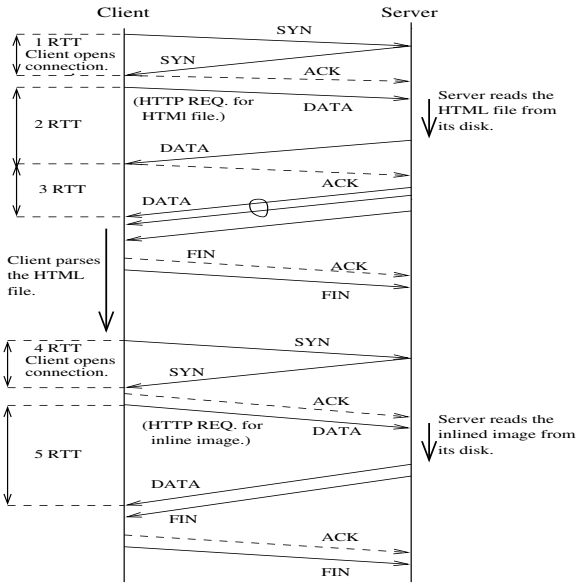

Figure 1: Packet exchange between a WWW client and server during a web page retrieval

WWW clients(browsers) use the Hypertext Transfer Protocol(HTTP) [1] to retrieve a WWW document(web page), stored on a remote computer on the Internet. The HTTP protocol is normally layered over the Transmission Control Protocol (TCP) [3]. A web page is written using the Hypertext Markup Language (HTML) [2], and stored as a plain text file. In such a HTML file, the textual content of the web page is written as plain ASCII text with HTML tags which determine the layout. Each inlined image[1] of the web page is encoded and stored as a separate image file, and HTML provides a mechanism by which the main HTML document representing the web page carries references to such image files at appropriate positions within its textual content. Whenever a user wishes to retrieve a web page, the client program is provided with the Uniform Resource Locator(URL)[2] of the corresponding HTML file. The client then requests the remote web server for the file, and upon retrieving it, parses the file to extract the URLs for any inlined images referenced therein. Separate requests are then made by the client for each inlined

[1]Inlined images are the photographs or icons that appear along with the text of a web page.

[2]An URL specifies the address of the object to be retrieved on the remote computers file system.

image of the web page.
Figure 1 illustrates the packet exchange between the WWW client and server during the retrieval of a web page with a single inlined image. In this figure all the packets are marked with their TCP packet types. Packets carrying data, are explicitly marked as DATA packets. The packets marked as ACK only carry acknowledgments and no data. The control packets, like the SYN and the FIN packets, are used to open and close the TCP connections respectively. These control packets usually do not carry any data. Figure 1 also illustrates the network round-trips incurred in retrieving the complete web page(the HTML file and the inlined image). The web page retrieval process can be summarized as follows:

1.) The client opens a TCP connection with the remote web server, which results in the exchange of SYN packets as a part of the TCP three way handshake procedure.

2.) The client transmits an HTTP request to the server to retrieve the HTML file. The server machine reads the file from its disk, and begins sending the data packets of the HTML file to the client. As illustrated in figure 1, the HTML file being large, doesnt fit into a single data packet. After sending the first data packet the server waits for an acknowledgment from the client before sending the next two packets. This is a direct consequence of the slow start mechanism adopted by the TCP protocol to reduce network congestion. Finally, after the entire file has been sent, the server closes the TCP connection by sending a FIN packet to the client.

3.) The client parses the retrieved HTML file to extract the URLs of inlined images, if any. In this example reference to a single inlined image is encountered in the HTML file. The client uses the corresponding inlined image URL, to open a new TCP connection with the server and retrieve the image file.

4.) The encoded image file gets transferred to the client following the same HTTP request-response sequence as was used to retrieve the main HTML file. If there are multiple inlined images referenced in the HTML file, the client opens separate TCP connections with the server for each inlined image, and then retrieves these inlined images, by sending separate HTTP requests on the respective connections. In the case of multiple inlined images, the TCP connection establishment for an inlined image, and the corresponding inlined image data transfer, need not wait for the previous inlined image to be fully retrieved. Some degree of parallelism is used by the client in retrieving the inlined images, but it is required that the main HTML file containing the URLs of the inlined images be fully retrieved before any TCP connection for inlined images of the web page can be opened.

The current HTTP protocol results in an inefficient interaction between the WWW client and the server. One of the major inefficiencies is the significant number of network round-trips incurred in retrieving the web page. In the example illustrated by figure 2-1, five round-trips are incurred in retrieving the complete web page data. Three round-trips for retrieving the main HTML file, and two round-trips for retrieving the single inlined image. These round-trips limit the earliest time by which the client can display the complete web page. In general, every object (HTML file or inlined image) retrieved by the client incurs at least two round-trips. One for the TCP connection establishment, and the other for the HTTP request-response sequence. The

round-trips illustrated in the figure become even more important when the client machine has to contend for the upstream channel to send the packet to the server as in the case of most of the broadband access networks. Local access delays and round-trip time of the access network will add to the basic round-trip time through the Internet and increase the overall round-trip time between the Client and the Server.
If we are to model the above HTTP transaction, the parameters of interest would be 1.)The TCP connection set-up time 2.)The Size of the request packets 3.)The size of the retrieved objects 4.)Time to parse the main HTML object 5.)Number of inline objects in an HTML document 6.) User think time in viewing a web page.

3 Previous Modeling Work

In the last few years, a number of Internet traffic modeling papers have appeared [5,6,7,9,10,11], which have primarily focussed on the aggregate characteristics of the traffic, based on the protocol used. It has been recently shown[11] that aggregate WWW traffic exhibits self similarity, which has been inferred from the underlying distributions of the WWW document sizes, the effect of user "think time" and superposition of many such transfers in a local area network. In [10], work has been done in deriving an empirical model for the document arrival process at an access link. This empirical model only characterizes the arrivals of the *HTTP document-requests* at an access link, ignoring the short TCP/IP control packets in between these document requests. Such short TCP control packets will be the major component of the upstream traffic presented by the WWW client source to the broadband access network. We believe that the source model proposed in [10], along with information on the requested object size is useful for testing resource reservation algorithms at the Head-End with a contention free upstream channel [3]. However absence of the short and frequent control packets in the source results in gross underestimation of the actual load on the contention channel of a cable modem when such a model is used for testing the performance of media access control protocols.
Mah[13] presents empirical statistics on some of the parameters discussed in this paper. However no modeling of these parameters is done. Our work reported in this paper is a more detailed study of the packet exchange in an HTTP transaction. We look at different transaction parameters (user dependant as well as network dependant), and then present analytical models for the measured distributions for each parameter. The models for the different parameters can then be used for actual packet generation.

[3]cable modems using the telephone line return path

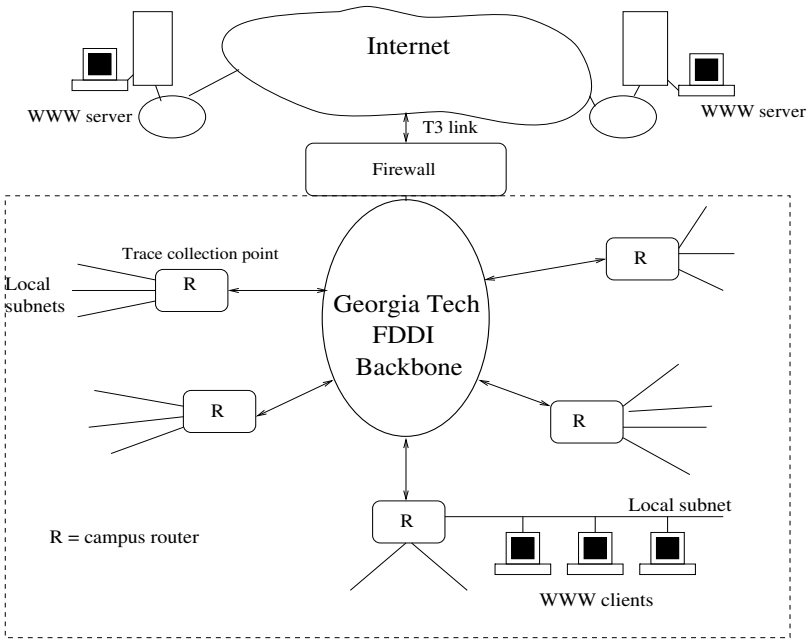

Figure 2: Setup for collecting trace data on WWW packets between clients on Georgia-Tech campus and web servers around the Internet.

4 Data Collection and Processing

Modeling of the WWW client behavior during a browsing session, requires statistical measurements on actual WWW traffic data. While related work[11] in the past, modified the code for the **mosaic** browser for application level data collection, the same was not feasible in our case at present, since **netscape** happens to be the most widely used browser nowadays, and the source code for is not yet readily available for modification. Thus we set up traces to collect WWW related packets on the local network, and later analyzed individual sessions from such traces .

4.1 Trace data collection

For our preliminary analysis, we collected WWW traffic data by monitoring the HTTP packets on the Georgia Tech FDDI backbone network. We ran the *tcpdump* packet capture utility on a *2 CPU(60 Mhz) Sparc20 workstation,* connected to the backbone, and used a HTTP packet filter (tcp port 80) to restrict our data collection to WWW related packets. The trace was collected on April 1st, 1997 between 11:00 am and 12:00 pm. Different Georgia Tech hosts spread across the campus, ranging from PCs to high-end workstations were involved in WWW browsing activity

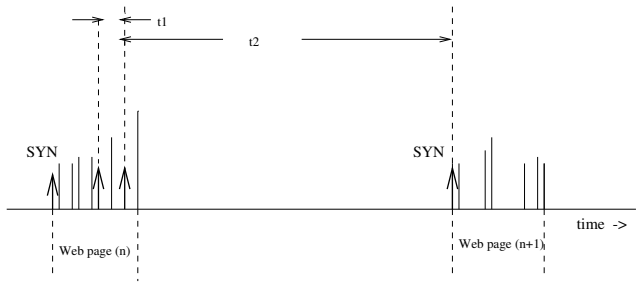

Figure 3: Interarrival times between the consecutive SYN packets during web browsing session.

during the trace collection period. The trace was filtered so that all clients were on Georgia Tech campus, thereby being close to the trace collection machine on the backbone. The WWW related packets recorded on the trace log file in real time were thus a complete record of the WWW browsing sessions between several local WWW clients running on campus machines and different remote web servers around the Internet [Figure 2]. The trace provided data on 11182 web page retrievals.

4.2 Trace data processing and the WWW client traffic analysis tool

The raw trace file provided by the tcpdump program was a real time record of the WWW related packets from different hosts on campus and remote web servers. Since many of the browsing sessions on the hosts were running in parallel at the time of trace data collection, packets pertaining to different web page retrievals were interleaved with one another. The off-line preprocessing of the trace file involved identifying the individual hosts which participated in browsing sessions, and then separating the interleaved packets of the raw trace file on a per host basis. We have developed a WWW client traffic analysis tool for this purpose, which after identifying the different participating hosts, scans the raw trace file and writes out all the packet interactions pertaining to a specific host, into a host file with a format suitable for further analysis. Such a host file thus contains a complete record of the packets that arrived/left a specific host on the campus, during the browsing session. However the packets pertaining to the different web page retrievals occur back to back in the host file, and we need to segment the host file into groups of packets, each group of packets representing the packet exchanges during a single web page retrieval. Our segmentation of the host file was based on the following observation.

The packets exchanged between a WWW client and remote web servers during a browsing session are temporally organized as shown in Figure 3. As discussed in section 2, a burst of packets (control,data) is exchanged between the WWW client, and the remote web server while the web page is being retrieved. This burst of packets is followed by an inactivity period during which

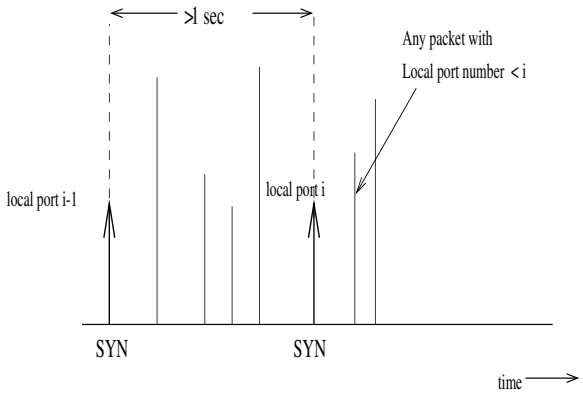

Figure 4: Possible error condition wherein two SYNs of the same web page have inter-arrival greater than 1 .

no packets are exchanged. This inactivity is due to the fact that the user is viewing the last retrieved web page, and hence no packets need to be exchanged at such times.
The different SYN packets of the same web page retrieval(burst period), have a small inter-arrival time as compared to the inter-arrival time between the last SYN packet of a burst period and the first SYN packet of the next burst period [Figure 3]. For our segmentation purpose we use a threshold of one second. The threshold of one second is based on the assumption that the user reaction time to a displayed web page is at least one second, and hence the first SYN packet of a new web page request cannot have an inter-arrival of less than 1 second with respect to the previous SYN. The PERL script we have developed for the segmentation, monitors the consecutive SYN packets in the host file to detect a condition wherein consecutive SYN packets have an inter-arrival greater than the threshold. Once such consecutive SYN packets are encountered, the later SYN packet is considered to be a likely first SYN packet of the next web page. Though not so common, due to some network problems, consecutive SYNs of the same web page retrieval may also be delayed by more than the threshold of 1 second as indicated in Figure 4.

The script thus performs a more detailed analysis of the host trace file in the vicinity of the suspected SYN packet. We use some peculiar characteristics of the WWW client-server interaction as discussed in section 2 to detect likely segmentation error conditions. Among the prominent error conditions checked are the possibility that the previous web page retrieval may not have ended at the time of detecting the suspected SYN packet. In such a case, some packets of the incomplete web page will be found after the suspected SYN packet and immediately identified by a lower local TCP port number Figure 4. If such a situation does occur, the script considers the suspected SYN packet as a part of the previous web page retrieval and proceeds with the scan for a suspicious SYN packet satisfying all the conditions for being the first SYN packet of a new web page retrieval. Once the start and the end of the overall transaction are detected in this fashion, different parameters of the transaction can be extracted.

5 Statistical Characterization of The Web Browsing Session

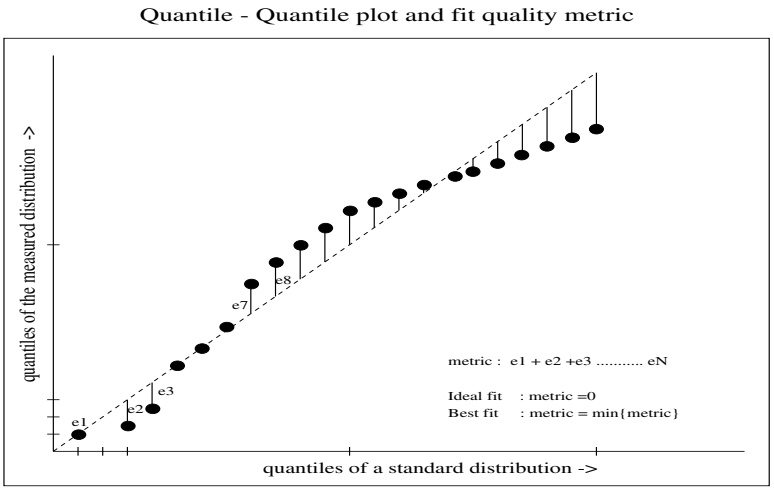

Figure 5: quantile-quantile plot of measured distribution with a standard probability distribution.

To derive a statistical model for the two-way HTTP packet transaction, we extracted information on the different transaction parameters discussed in section 2 from the trace file. Once empirical distributions for the individual parameters were obtained, we compared each distribution with different standard probability distributions and selected the best fit. We used quantile-quantile plots in our study. Quantiles of the measured distribution were plotted against the corresponding quantile values for a standard probability distribution with the same mean and variance. A perfect fit is the standard probability distribution whose quantile values are equal to those of the measured distribution, resulting in a straight line through the origin with slope equal to one. The metric we used to compare the fits is a sum of the deviations from the perfect values at each quantile point as shown in Figure 5. The best fit in this case is the standard distribution which minimizes this metric.

5.1 Packet exchange parameters

TCP connection setup time $T_{connect}$

This parameter is useful in modeling the basic round trip time (rtt) between the Client and a remote web Server on the Internet. To note the connection setup time, the duration between the SYN packet sent by the Client and the corresponding SYN packet received from the Server

was measured. At times when the SYN packet from the client was lost in the network, or the Server could not accept the connection for a while, the Client timed out and sent the SYN packet again. However since the connection set-up time was recorded from the first SYN packet sent by the client for that connection, such connections which were finally established after timeouts and retransmissions resulted in connection set-up times extending to a few seconds. These abnormal delays due to initial connection establishment failures usually above 500ms have been treated as outliers and filtered out. The histogram for the filtered rtt values is shown in Figure 6.

As illustrated in the histogram, most of the time the connection setup requires a few hundred

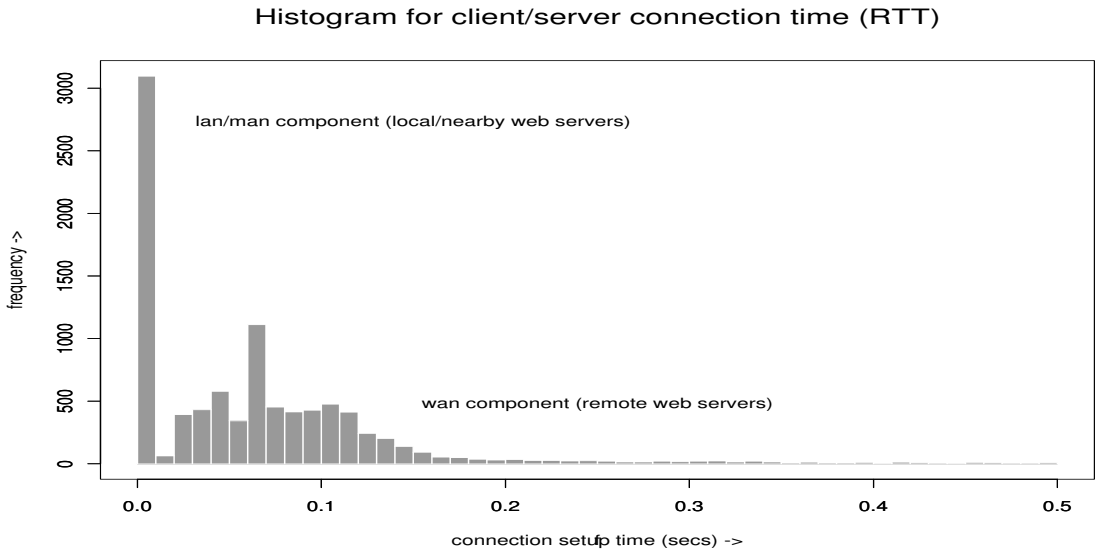

Figure 6: Histogram for the TCP connection establishment delay (rtt)

milliseconds depending upon the physical distance of the client machine from the web server and the queuing delays in the intermediate routers. The histogram illustrates a mixture of two distributions, one for the local rtt component and the other for wide area rtt component. The local rtt component extends from 0 - 0.0025 secs the wide area component extending over the remaining 0.0025 to 0.5 secs range. The probabilities of the rtt being from the local component and wide area component are .2964 and .6493 respectively with the rest being the filtered rtts. Zooming in on the histogram for the local rtt component, we see a distinct cluster of delays pertaining to sessions with nearby web servers. The distribution as illustrated in Figure 8 is fairly symmetric with a mean of 0.001257 secs, median of 0.001252 secs and a standard deviation of 0.0003032. A normal distribution with same mean and variance gives the best fit. The quantile-quantile plot for a normal approximation is shown in Figure 9

For the wide area rtt component (0.0025 to 0.5 secs) shown in Figure 10 we have observed a mean of 0.09392 secs, median of 0.0733 secs with a standard deviation of 0.0727. Some of the higher end delays are a result of connecting with distant web servers having high round-trip propagation delays and/or congestion and queieng in the intermediate routers and/or connecting with heavily loaded web servers taking time to send the connection response packet. The best fitting distribution for the measured wide area rtts is a log normal distribution, the corresponding

quantile-quantile plot being illustrated in Figure 11.

HTTP Request Packet Size *Reqsize*

The HTTP request-packet typically carries the address of the HTTP object to be retrieved from the remote web-servers file system. This request also contains information used for authentication of the user and for specifying data formats understood and accepted by the Client software sending this request packet. As shown in Figure 12 most of the request packets are around 300 bytes (including 40 bytes for the tcp/ip headers). Occasionally the request size may show higher packet sizes, which can be online *forms* data submitted by the user. Figure 13 shows the quantile-quantile plot for the HTTP request packet size distribution, with its best fit, a log normal distribution.

TCP Data Segments *Numsegs*

The actual file sizes of WWW documents have been shown to belong to a heavy tailed distribution like a Pareto distribution in [11] . For the purpose of packet exchange we measure the number of data packets (tcp segments) that were sent by the server to the client in response to a document request. This helps in estimating the number of acks sent out by the client. We have observed that both the main object as well as the inline object, have similar distributions for the number of tcp data segments transfered from server to the client. The mean number of segments for the main document is 8.3 [9] with a median value of 4. The mean number of segments for the inline object is 7.27 [8] with a median of 3. The inlined objects tend to be slightly smaller as they are usually compressed image files used for the icons in a web page as compared to the uncompressed ASCII HTML file (main object). The histogram for the number of data segments for the main object and the quantile-quantile plot with a log normal distribution are illustrated in Figures 14 and 15 respectively.

Parsing Time T_{parse}

Once the main HTML document is received by the client, it needs to parse the file before opening more tcp connections with the server for inline objects of that web page if any. The histogram for the parsing time, Figure 16, shows a mean of 148 ms and a median of 64.2ms. However the actual parsing time can vary widely from a few microseconds to a few hundred milliseconds based on the capabilities of the Client machine and the size of the main HTML file being parsed. The best fitting distribution for the file parse time is a gamma distribution, the quantile-quantile plot for the same being shown in Figure 17

Number of Inline Objects $Numinline$

The measured distribution for the number of inlined objects (Figure 18) shows a mean of 1.901 [2] inline objects per web page with a median of 0. However we would like to add that this distribution is highly dependent on the content of the web page being retrieved. The type of web pages typically retrieved in a research community is more likely to be text oriented objects with very few inline images. The greater the multimedia content[4] in the web page, the more will be the number of inline objects in the same web page.
A gamma distribution with the same mean and variance gives the best continuous fit for the measured data. Figure 19 illustrates the corresponding quantile-quantile plot.

Web Page View Time (User think time)T_{view}

Once a web page is retrieved by the client, the user spends some time in viewing the page before clicking for the next web page. This time period which we call as the user think time, shows a high variance and is the only parameter that is directly influenced by the user and not the network. For the histogram on web page view time in Figure 20, we filtered out view times that extended beyond 3 minutes, as these off-times are produced by the user who typically fetches a web page and after viewing it, does not browse for a long time while working on some other work. The measured distribution for the user think time shows a mean value of 21.04 secs and a median of 9.84 secs. The distribution is best approximated by a Weibull distribution.

Upstream IP packet size

Besides the parameters that directly influence the HTTP transaction, we noted some general statistics on the client traffic packet flow in the upstream direction. Out of the statistics gathered, one interesting observation was the size of the outgoing packets from the WWW Client. As would be expected for the highly asymmetrical web browsing session, the histogram for the IP packet size shown in Figure 22 indicates that most of the time, the IP packet size leaving the WWW client in the upstream direction is 40 bytes due to the frequent TCP/IP control packets like the SYN/FIN/ACK packets. Such individual control packets can easily fit into a single upstream ATM payload of 48 bytes and hence carrying such short packets should not be a problem for ATM friendly MAC encapsulation layers envisioned for the broadband access networks.

[4] photographs, bullets, icons, audio/video clips

6 WWW Traffic Generation

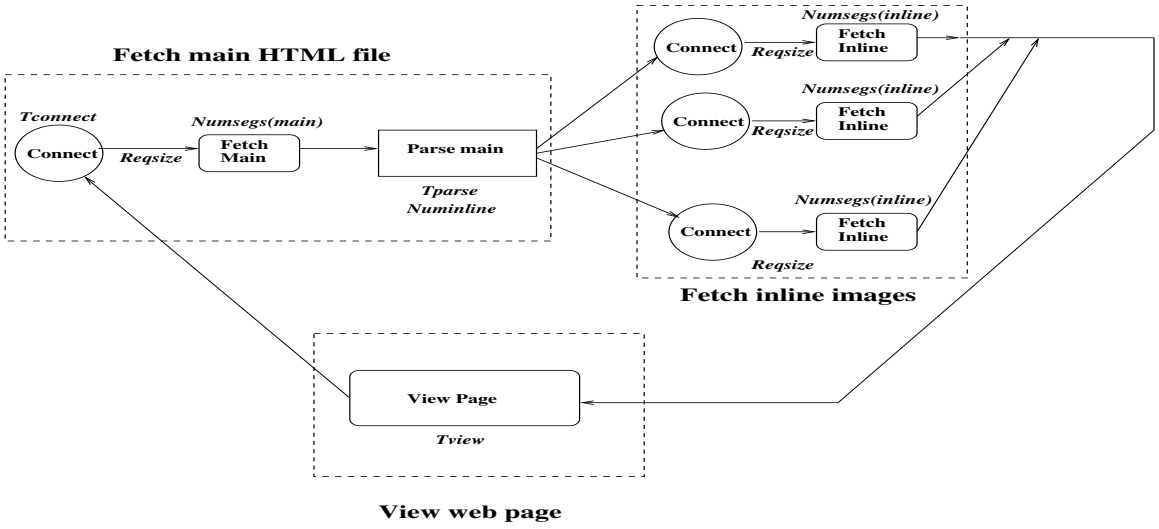

Figure 7: State transition model for a WWW packet generator

Parameter	Mean	Median	Stddev	Model Distribution
Connection setup(rtt) time	0.001257 secs	0.001252 secs	0.0003032	Local component: *Normal*
$T_{connect}$	0.09392 secs	0.0733 secs	0.0727	Wide area component:*Log normal*
$Reqsize$	324 bytes	299 bytes	84.9	*Log normal*
$Numsegs(main)$	8.3 [9]	4	11.82	*Log normal*
$Numsegs(inline)$	7.26 [8]	3	16.52	
T_{parse}	0.1484 secs	0.0642 secs	0.2102	*gamma*
$Numinline$	1.902 [2]	0	3.965	*gamma*
$T_view(think)$	21.0396 secs	9.8405 secs	29.0784	*Wei bull*

Table 1: Summary of models for the HTTP transaction parameters

The state transition diagram of a synthetic WWW traffic generator that could use our transaction parameter models is illustrated in Figure 7 with the models for the parameters summarized in the table below. The generator simulates the traffic which a WWW browser would present to an access network. The generator is an ON-OFF type bursty source, with the packet generation pertaining to the actual web page retrieval being simulated in the ON state, the OFF state representing the longer period of silence after web page retrieval for viewing the page. The ON state can be further split into the simulation of main document retrieval and inline object retrievals. The random numbers required for simulating the fetching of the main HTML file are, $T_{connect}$ for simulating the connection establishment delay/round-trip time between the client and web server, $Reqsize$ for deciding on the size of the HTTP request packet to be generated,$Documentsize(main)$ or $Numsegs(main)$ to decide on the ack packet generation during fetching of the main HTML file, and $Tparse$ for simulating the HTML-file parsing time. Subsequently, random number $Numinline$ indicates the number of inline file retrievals

to be simulated, each of which involve the same sequence of events as the HTML-file retrieval. Finally the time spent by the user in viewing the complete web page (no packets generated), is simulated by fetching a random number T_{view} and generating no packets for this time period. On the expiration of this view time, another web page retrieval begins again.
The advantage of splitting the packet generation into such separate states is that, we now have flexibility in controlling the overall packet generation process, and predicting the performance under different user/network specific scenarios. We can study the impact of different parameters by varying the arguments used in the analytical models(standard probability distributions) for these parameters.

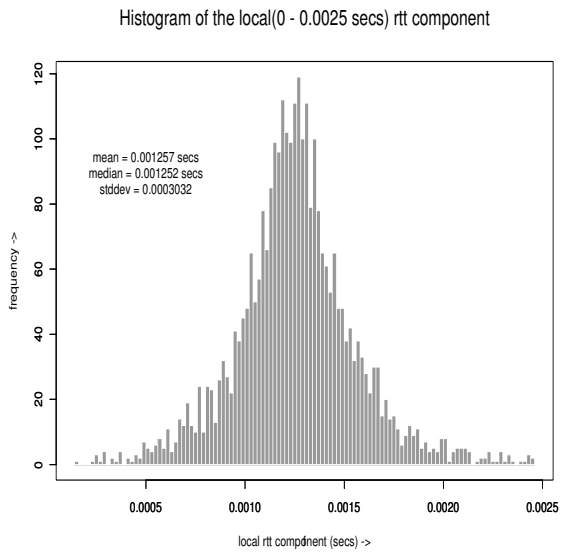

Figure 8: Histogram for the local rtt component

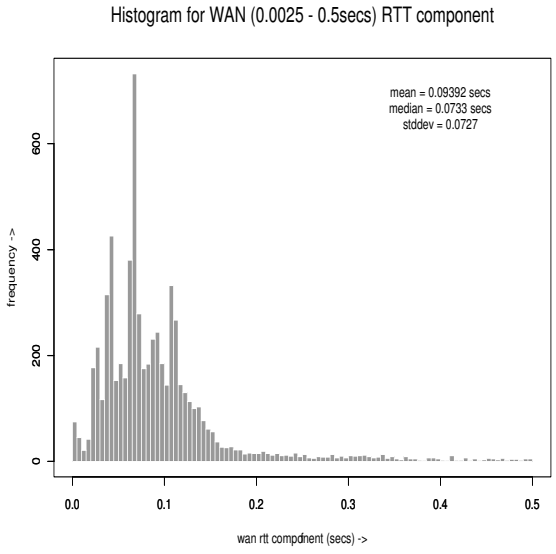

Figure 10: Histogram of the wide area rtt component

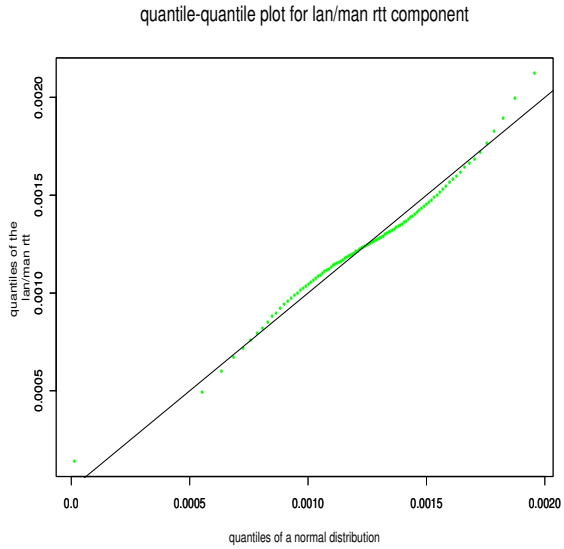

Figure 9: qqplot of the local rtt component with a normal distribution

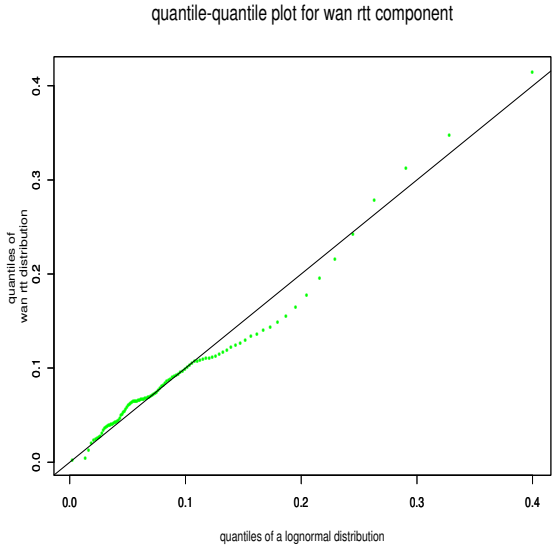

Figure 11: qqplot of the wide area rtt component with a log normal distribution

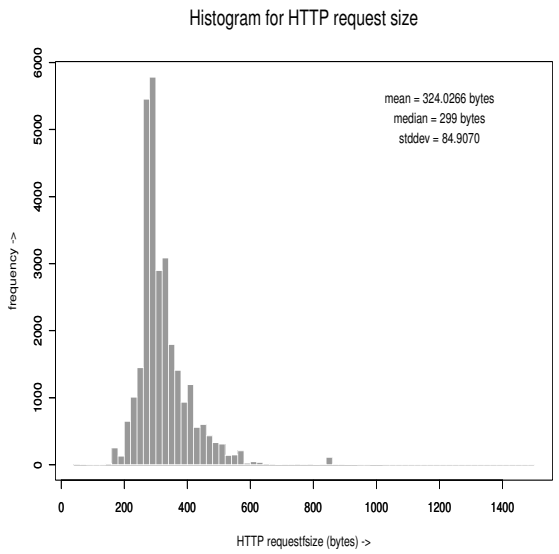

Figure 12: Histogram of the request packet size

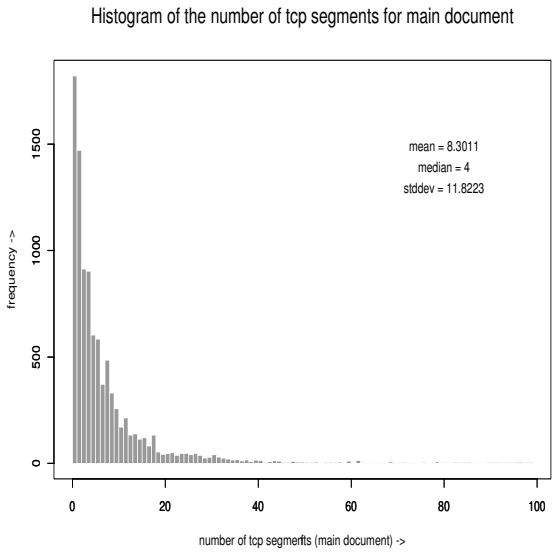

Figure 14: Histogram for the number of data segments transferred for the main HTML file

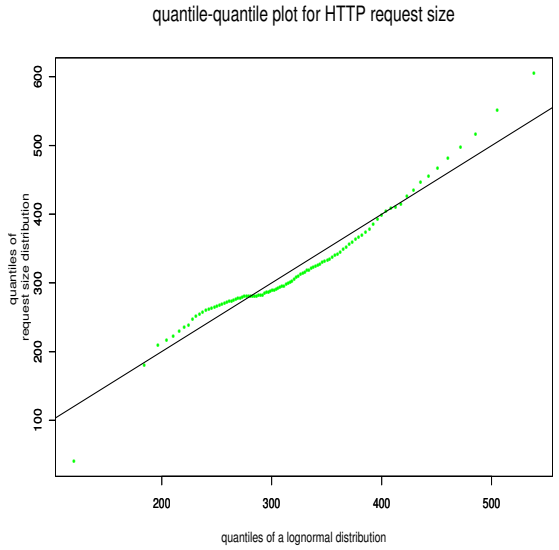

Figure 13: qqplot for the request packet size with a lognormal distribution

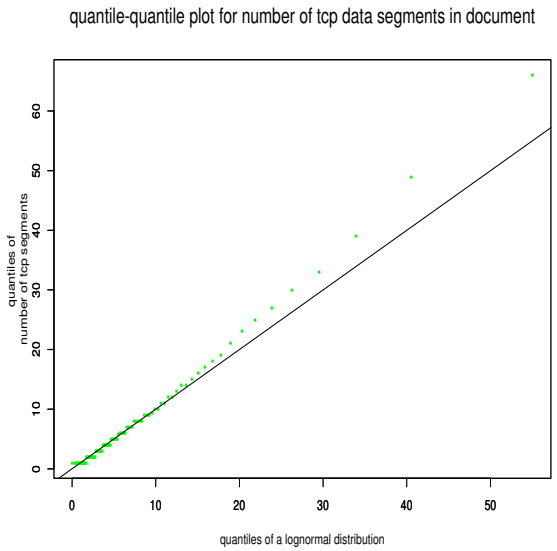

Figure 15: qqplot for the number of data segments with a log normal distribution

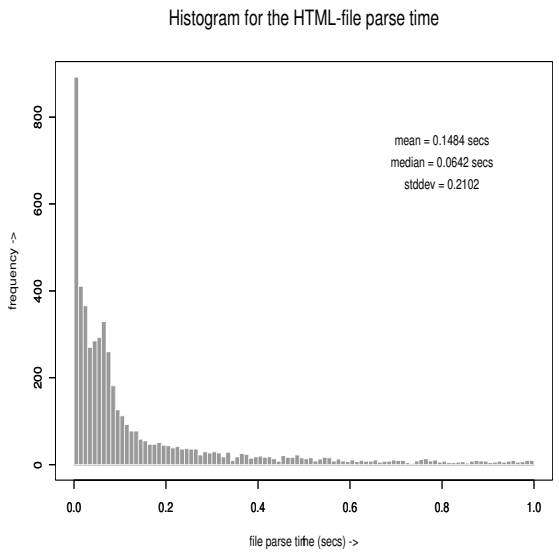

Figure 16: Histogram for the time to parse a HTML file

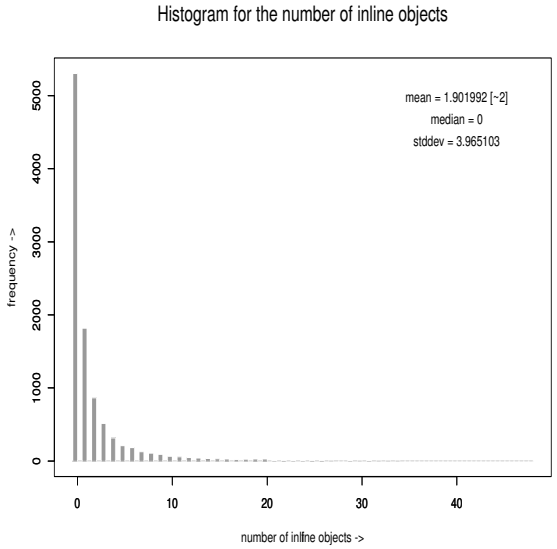

Figure 18: Histogram for the number of inline objects in a web page

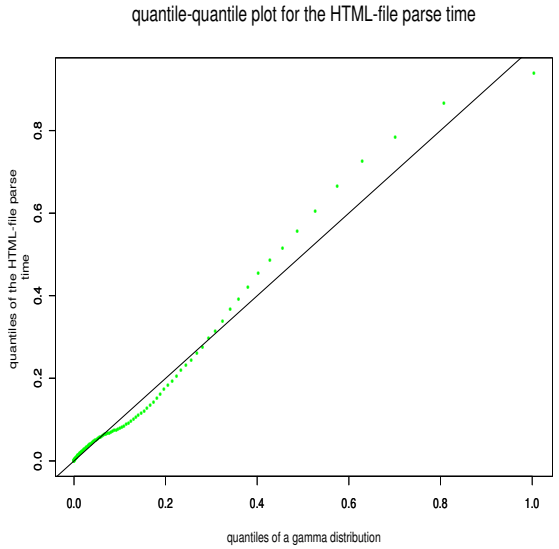

Figure 17: qqplot for the file parse time with a gamma distribution

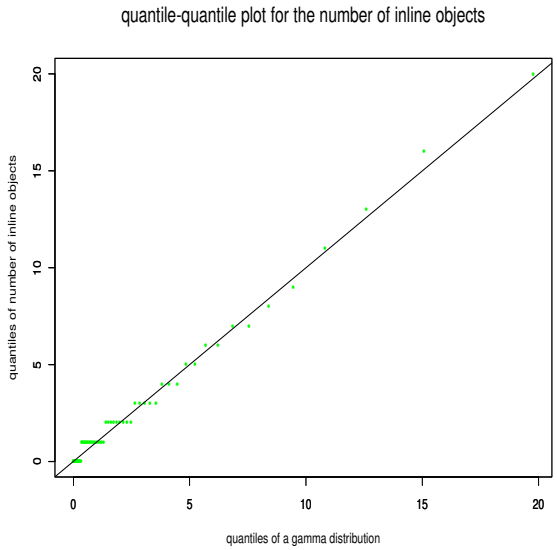

Figure 19: qqplot for the number of inline objects in a web page with a gamma distribution

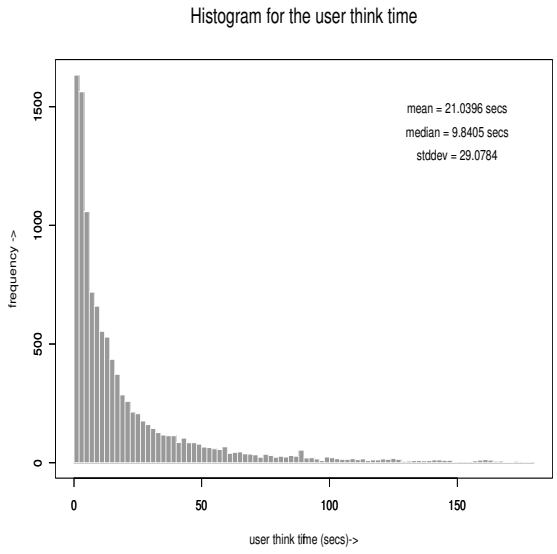

Figure 20: Histogram for the time spent in viewing a web page

Figure 21: qqplot for the web page view time with a Weibull distribution

Figure 22: Histogram for the size of IP packets presented to the access layer in the upstream direction during WWW browsing session

7 Conclusion and future work

In this paper we have presented different statistics on the packet exchange during the WWW browsing session. Each of the measured distributions have been approximated by best-fitting standard probability distributions. The advantage of using analytical models (standard distributions) for the measured distributions is that the parameter can now be represented in a more compact form using mathematical expressions for the corresponding density function. The resulting expressions for the parameters directly lend themselves to the development of a WWW traffic generator. The advantage of this traffic generator is that we have more flexibility in tuning the overall packet generation process by tuning the parameters of the specific underlying distribution. Such tuning would account for external factors like congestion in the Internet, access link bandwidth, time/day of the week and so on. A number of interesting issues are raised by our statistical characterization of the HTTP transaction. In most of the MAC layer proposals for cable modem access networks , the MAC layer in the cable modem has to make an upstream reservation for a slot in which to transmit the higher layer packet. This added request-grant round trip delay is not much of a problem if the request will be for a significant number of upstream slots. For the web traffic, each packet will fit into a single upstream slot, and unless smart granting/direct channel access schemes are used each short packet will give rise to a request packet upstream, and will be further delayed by the access delay of the MAC layer. The delay of each packet adds up and would affect the fastest time by which a web page can be fully displayed at the subscriber terminal. As discussed earlier, piggy-backing of new requests with the transmission of earlier packets will not help as the new packet is not available when the previous packet leaves the modem. Another conclusion to be drawn is that, concatenation of upstream data slots to reduce MAC overhead, will not help much for web browsing upstream traffic which occurs as primarily solitary cells.

References

[1] T.Berners-Lee, " Hypertext Transfer Protocol (HTTP) ", *Internet Draft draft-ietf-iiir-http-00.txt*, IETF, Nov. 1993.

[2] T.Berners-Lee and D. W. Connoly. "Hypertext Markup Language Specification - 2.0". *Internet Draft draft-ietf-html-spec-00.txt*, IETF, Nov. 1994.

[3] Jon B. Postel, "Transmission Control Protocol", *RFC 793, Network Information Center*, SRI International, Sept. 1981.

[4] V. Jacobson, C. Leres, and S. McCanne, "tcpdump", available via anonymous ftp to *ftp.ee.lbl.gov*, June 1989.

[5] Vern Paxson, "Empirically derived analytical models of wide area TCP connections". *IEEE/ACM Transactions on Networking* 2(4):316-336, Aug. 1994

[6] S. Heimlich, "Traffic characterization of the NSFNET national backbone" in *Proc. 1990 Winter USENIX Conf.*, Washington,DC

[7] James Pitkow and Margaret Recker, "A simple yet robust caching algorithm based on dynamic access patterns". *In Elect proceedings of second WWW conference*, 1994.

[8] Merit Network Inc. NSF network statistics. Available at *ftp://nis.nsf.net/statistics/nsfnet/* December 1994.

[9] H. H. Lee and C. K. UN, "A study of On-Off Characteristics of Conversational Speech",*IEEE transactions on Communications*, Vol. Com-34,No.6,June 1996.

[10] Shuang Deng, "Empirical model of WWW document arrivals at access link",(paper to be presented by the author at IEEE ICC'96).

[11] Mark E. Crovella and Azer Bestavros, "Self Similarity in World Wide Web Traffic, Evidence and Possible Causes", *(to appear in the 1996 ACM SIGMETRICS International Conference on Measurement and Modelling of Computer Systems, May 1996).*

[12] Sabyasachi Basu,Amarnath Mukherjee, and Steve Klivansky, "Time Series Models for Internet Traffic", Georgia Tech-College of Computing Technical Report,*GIT-CC-95-27.*

[13] Bruce Mah, "An Empirical Model of HTTP Network Traffic",*The Tenet Group, Computer Science Division, University of California at Berkeley*, submitted to INFOCOM '97.

[14] D. Sala and John Limb, "A Protocol for Efficient Transfer of Data over Fiber/Cable Systems", *Proc. of INFOCOM'96, pp. 904-911*, San Francisco, CA, March 24-28, 1996.