Web traffic modeling and performance comparison between HTTP1.0 and HTTP1.1

Z. Liu, N. Niclausse, C. Jalpa-Villanueva

INRIA Centre Sophia Antipolis
2004 Route des Lucioles
B.P. 93, 06902 Sophia Antipolis, France

Abstract

In this work we propose a stochastic model for describing the Web traffic at request level. We concentrate on the HTTP requests to the same Web server and we are interested in the typical behavior of the clients' hypertext navigation. The mathematical model is defined by a stochastic marked point process which describes when clients arrive and how they browse the server. We have developed a benchmark tool of Web servers: WAGON. It comprises a generator of Web traffic, a robot which sends and analyzes requests and a monitoring tool. It can generate different types of traffic requests, which, in turn, can be sent out to the server from different machines with different (simulated) delays and bandwidth constraints. Using such a tool we have carried out experiments and compared the performance between HTTP1.0 and HTTP1.1 under different configurations. Our results indicate that HTTP1.1 almost always outperforms HTTP1.0, and this is true even if the latter uses more parallel connections and slightly larger bandwidth. The difference in request response time becomes more significant when the network is heavily loaded. The response time of clients using HTTP1.0 with a single connection could be ten times larger than those using HTTP1.1.

Key words: Web traffic modeling, Heavy tail distribution, Long range dependence, HTTP, Web hypertext.

1 Introduction

The World Wide Web has become a major way of publication and information search on the Internet during the last several years. It is now the dominant source of the Internet traffic. The exponential increase of the number of servers and of the number of users causes performance problems of access to Web objects, due to the saturation of Web servers and of the communication network.

One of the main preoccupations of Web server administrators is to propose a fast and reliable service to satisfy their clients. The tremendous success of the Web makes this task difficult to accomplish. Indeed, not only is it necessary to provide an efficient service for a given time instant, but also it is required to anticipate traffic growth in order to maintain the quality of service for short or medium term.

It is thus important to understand the statistical properties of the Web traffic and to develop appropriate traffic models for the performance evaluation of Web servers and Web applications.

SURGE [1] is a benchmark tool that tries to imitate a stream of HTTP requests originating from a fixed population of web users. A user is modeled by an ON/OFF process (User Equivalent) who during the ON period makes requests for Web files and during the OFF period lies idle. Within an ON period there are active OFF time periods corresponding to the time between transfer of components of a page. Inactive OFF time periods correspond to the user think time. The workload is generated using an analytic approach to capture properties observed in real web workloads concerning file sizes (what is stored in the file system), request sizes (what is transferred over the network from the server), file popularity, and temporal locality. Distributional models for the file size, the active OFF times, and the embedded references, were developed using a client trace data set.

Several other Web benchmarks exist [10,9,6] which periodically generate simple HTTP requests to a small set of pages. In hbench:Web [5], the workload is generated according to empirical distributions based on server log files.

In our work, we propose a stochastic model for describing the Web traffic at request level. We concentrate on the HTTP requests to the same Web server and we are interested in the typical behavior of the clients' hypertext navigation. The mathematical model is defined by a stochastic marked point process which describes when clients arrive and how they browse the server.

HTTP request stream is dependant on both the protocol (depending on the use of parallel connexions or persistent one) and on the navigator (graphical or textual borwser for instance). What is typical of Web users behaviour is hypertext browsing (clic on links, read pages, and so on). Therefore, our goal was to design the protocol and implementation independant patterns of Web trafic, browsing behaviour. Other components (server contents and embedded objects requets) is left to the benchmarking tool. Our model focus on users behaviour while they browse a server: the number of clicks and the time elapsed between eahc clic.

We have developed a benchmark tool of Web servers: WAGON (Web trAffic GeneratOr and beNchmark). It comprises a generator of Web traffic, a robot which sends and analyzes requests and a monitoring tool. It can generate different types of traffic requests, which, in turn, can be sent out to the server from different machines with different (simulated) delays and bandwidths. Using such a tool we have carried out experiments and compared the performance between HTTP1.0 and HTTP1.1 under different configurations.

Our results show in particular that HTTP1.1 almost always outperform HTTP1.0, and this is true even if the latter uses more parallel connections and slightly larger bandwidth. The difference becomes more significant when the pipeline mechanism is used in HTTP1.1 and when the propagation delay is large.

The paper is organized as follows. In section 2 we describe the traffic model we proposed and we report the statistical analysis results. In section 3 we present the experimental results obtained using WAGON.

2 Traffic Model and Statistical Analysis

2.1 Traffic Model

We propose a stochastic model of HTTP traffic. We confine ourselves to HTTP requests to the same Web server. We are interested in the typical behavior of clients traversing the hypertext of a Web server, namely, when they arrive and how requests are generated. This can actually be well described by the notion of session: a sequence of clicks of a client on the hyperlinks of the same server.

The traffic model we propose can be described by a stochastic marked point process. The arrival times correspond to the beginning times of sessions. Each arrival is associated with the following random variables (the marks):

- The number of clicks during the session;
- The times elapsed between successive clicks;
- The addresses of the successively required pages, i.e., the addresses of the first page and the successively clicked hyperlinks.

Note that in this traffic model for benchmarks, the embedded objects (icons, images, animations, audio, etc.) are not specified and are requested automatically by clients, by parsing HTML files.

It turns out that the arrival process of sessions is well described by a Poisson process, see results below. This permits a simple way to parameterize the traffic intensity. The distribution functions of the number of clicks and of the interclick times most often belong to the classes of Pareto, lognormal, Inverse Gaussian and Weibull distributions. The page addresses are samples of the weighted directed graph representing the hypertext structure of the server. These weights correspond to the visit frequencies and the routing probabilities (i.e. the probabilities of following the hyperlinks).

2.2 Statistical Analysis Results

Statistical analyses have been carried out on different log files, in particular, those of the Web servers at W3C[1], INRIA[2] and www.clark.net[3]. Some characteristics of these server traces are in Table 1.

Table 1
Characteristics of the servers www.w3.org, www.inria.fr, www.clark.net

	www.w3.org	www.inria.fr	www.clark.net
Time period:	Feb 97	Oct 96	Sep 95
Duration:	24h	24h	18h
Total number of requests:	275000	50000	150000
Total number of pages:	4726	7773	9294
Average page size:	23kB	15kB	13kB

The results of the statistical analyses and the parameter identifications are summarized in Table 2.

Previous results in the literature concerning the analysis of HTTP and IP traffic have rejected the Poisson hypothesis, see for example [8,2]. However, if one looks at the arrival process of sessions instead of the arrival process of requests or packets, the Poisson assumption turns out to be valid.

The numbers of clicks of sessions form a renewal process whose increments have a heavy-tail distribution. For the traces we analyzed, this heavy-tail distribution is either inverse Gaussian or Pareto.

Our analyses indicate that the interclick times are positively correlated, although the coefficient of auto-correlation is very small. There also seems to be long range dependence with a small Hurst parameter, see figure 1.

[1] http://www.w3.org/
[2] http://www.inria.fr/
[3] http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html

Table 2
Distributions of servers www.w3.org, www.inria.fr, www.clark.net

Random Variable	**Matching with Hypothesized Distribution**	**Mean**	**Standard Deviation**
Distributions for the server www.w3.org			
Session arrivals	Poisson process ($\lambda = 0.39$)	2.5s	2.7s
Number of clicks per session	Pareto ($a = 0.748; \beta = 0.807$)	3.1	4
Interclick time	LogNormal ($m = 2.84; \sigma = 1.68$)	63s	138s
Distributions for the server www.inria.fr			
Session arrivals	Poisson process ($\lambda = 0.037$)	27s	31s
Number of clicks per session	Inverse Gaussian ($\mu = 5.96; \sigma = 3$)	6	11.5
Interclick time	LogNormal ($m = 3.26; \sigma = 1.36$)	68s	121s
Distributions for the server www.clark.net			
Session arrivals	Poisson process ($\lambda = 0.21$)	4.7s	5s
Number of clicks per session	Pareto ($a = 0.94; \beta = 1.16$)	3.3	4.4
Interclick time	LogNormal ($m = 3.8; \sigma = 1.3$)	101s	152s

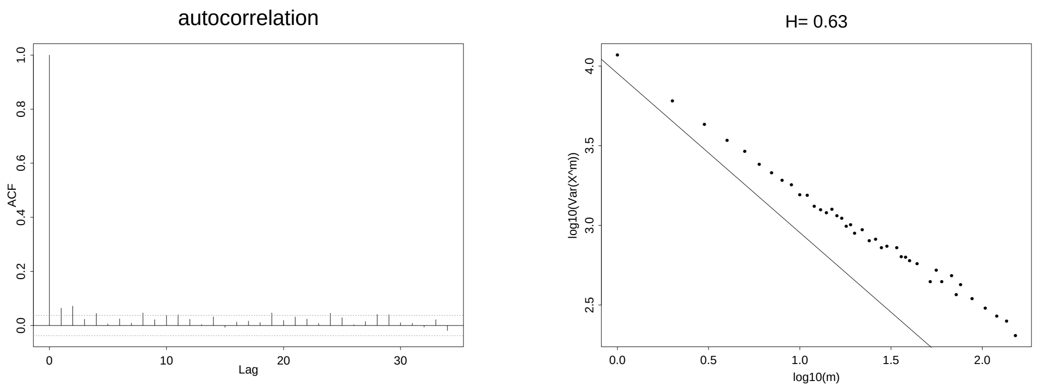

Fig. 1. Dependence analysis of interclick times

3 Experimental Results

3.1 WAGON

We have developed a benchmark tool of Web servers: WAGON. It comprises a generator of Web traffic, a robot which sends and analyzes requests and a monitoring tool. It can generate different types of traffic requests, which, in turn, can be sent out to the server from different machines with different (simulated) delays and bandwidths. For this, user can specify server features (server name or IP address, home page address of the server, server log files, etc.), client features (laws and parameters of the random variables in the traffic model; transport protocols and parameters used), and experiment configuration (local client caching, sampling rate, client machines and bandwidth constraint and routing delay on these machines, etc.).

During a benchmark, various performance measures can be obtained and observed on-line through the monitoring tool of WAGON. Of particular interest are request latency and user perceived throughput.

3.2 Long-range dependence and the traffic model

In our traffic model, we use mutually independent random variables for the marks of the point process. However, the resulting traffic is long-range dependent, and thus asymptotically self-similar, as illustrated in figure 2.

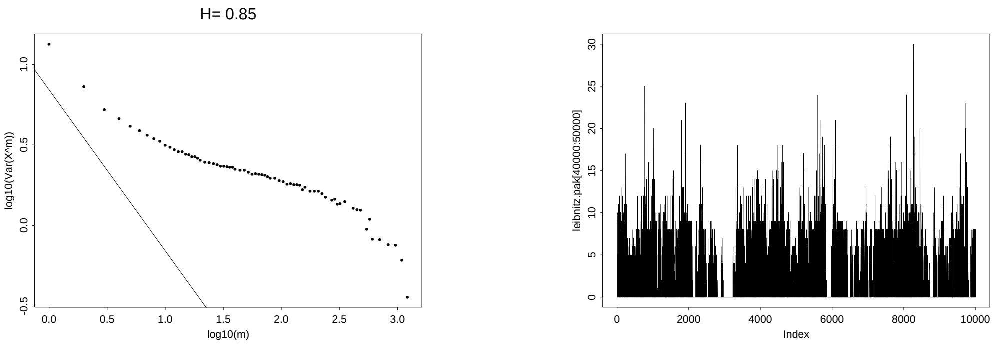

Fig. 2. Variance-time plot of the observed traffic generated by WAGON

This result is actually not really surprising. As is already shown in [3], when one superpose an infinitely many on-off sources where the "on" times have a heavy-tail distribution, the resulting process is long-range dependent.

3.3 Performance Comparison between HTTP1.0 and HTTP1.1

At the time HTTP1.1 was proposed, a comparison study between HTTP1.0 and HTTP1.1 was carried out [7]. The effect of persistent connections, pipelining, document compression, bandwidth, and latency, on the performance was investigated using a single Web page containing 42 embedded images. A work to characterize Web response time is presented in [4]. Four proxy logs files are used, they are replayed using a URL re-player multi-process program. The workload is replayed by reading a log file of URLs, sending HTTP requests, and timing the transfer. Ten experiments were done to investigate the effects of proxy caching, network bandwidth, traffic load, and persistent connections. The periodicity of the response times was also studied.

We investigate the following four types of connections: modems, T1 links, WAN (transatlantic) and satellite connections. In order to simulate such connections, we put the delay and bandwidth constraints as in table 3.

Table 3
Delay and bandwidth constraints

	delay	bandwidth
modem	300 ms	33kb/s
T1	40ms	1.5Mb/s
WAN	180ms	100kb/s
satellite	500ms	2Mb/s

The parameters used for generating Web traffic is illustrated in table 4.

Table 4
Clients parameters used in the experimentations

Variable	**Law**	**Mean**	**Std deviation**
Session arrival	Poisson Process ($0.0005 < \lambda < 0.03$)		
Number of clicks	Inverse Gaussian ($\mu = 5; \lambda = 3$)	5	1.28
interclick time	LogNormal ($m = 3; \sigma = 1.1$)	36.8s	56.4s

We have chosen two types of Web server contents which are both a subset of the Web server INRIA [4] . Server 1 consists of many large pages (mostly PostScript files). Server 2 consists of HTML files in majority with more embedded images. Their characteristics are summarized in table 5.

Table 5
Characteristics of the Web servers used in the experimentations

	server1	server2
Total number of pages:	778	2353
Average size:	58kB	6.5kB

.

The Apache server (Version 1.3) [5] is used to serve the requests. The server accepts both HTTP1.0 and HTTP1.1 requests. The time out for persistent connections is set to be 30 seconds.

The comparison results are illustrated in figures 3 and 4. We considered four configurations of HTTP control parameters: HTTP1.0 with 1 connection per

[4] http://www.inria.fr
[5] http://www.apache.org

client, HTTP1.0 with 4 parallel connections per client, HTTP1.1 with 1 connection per client without pipeline, HTTP1.1 with 1 connection per client with pipeline. Recall that in the current implementation, the bandwidth constraint is applied per connection instead of per client. Thus, a client using HTTP1.0 with 4 parallel connections can actually have larger available bandwidth.

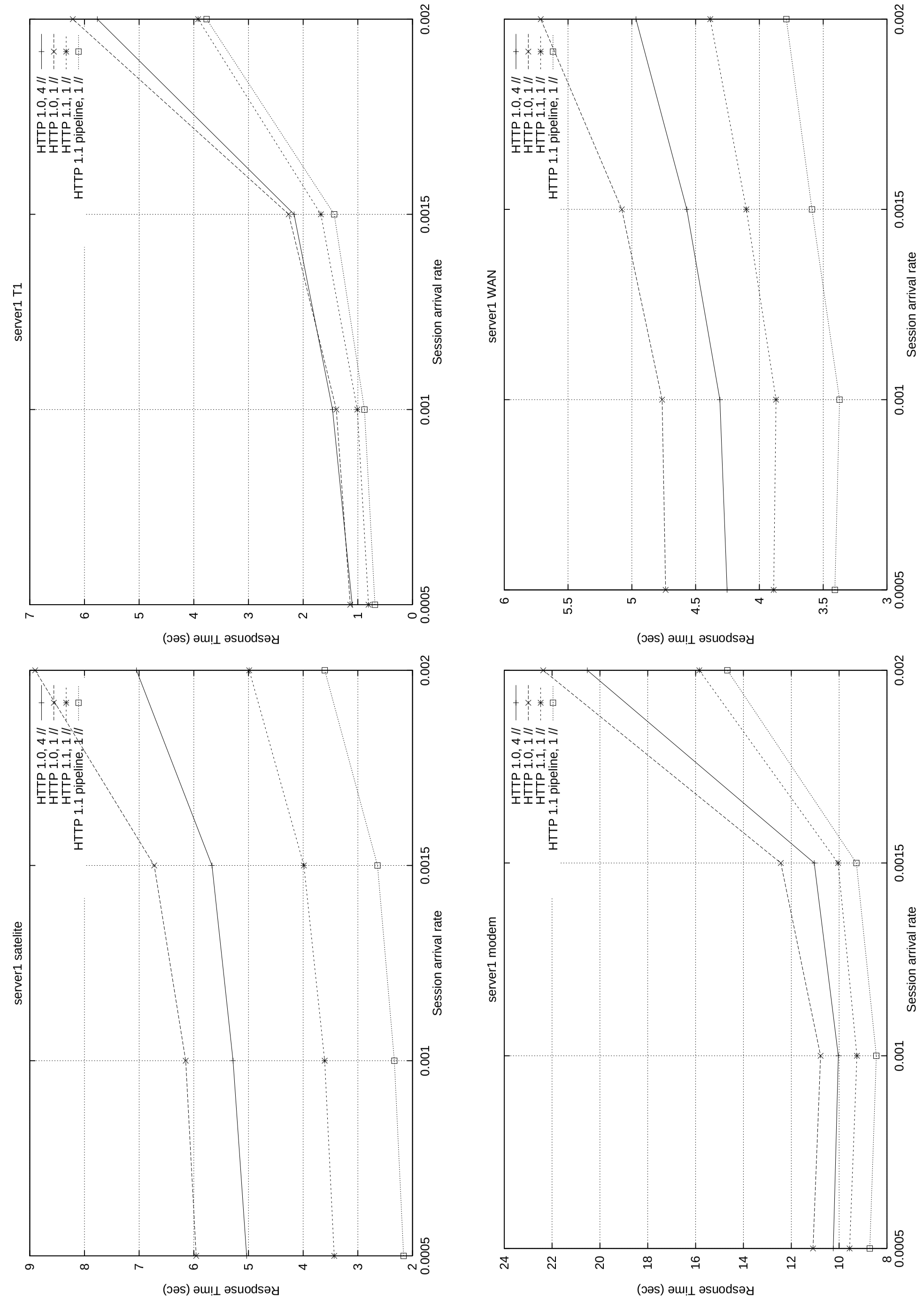

Fig. 3. Performance comparison between HTTP1.0 and HTTP1.1 on server 1

These results indicate that HTTP1.1 almost always outperforms HTTP1.0,

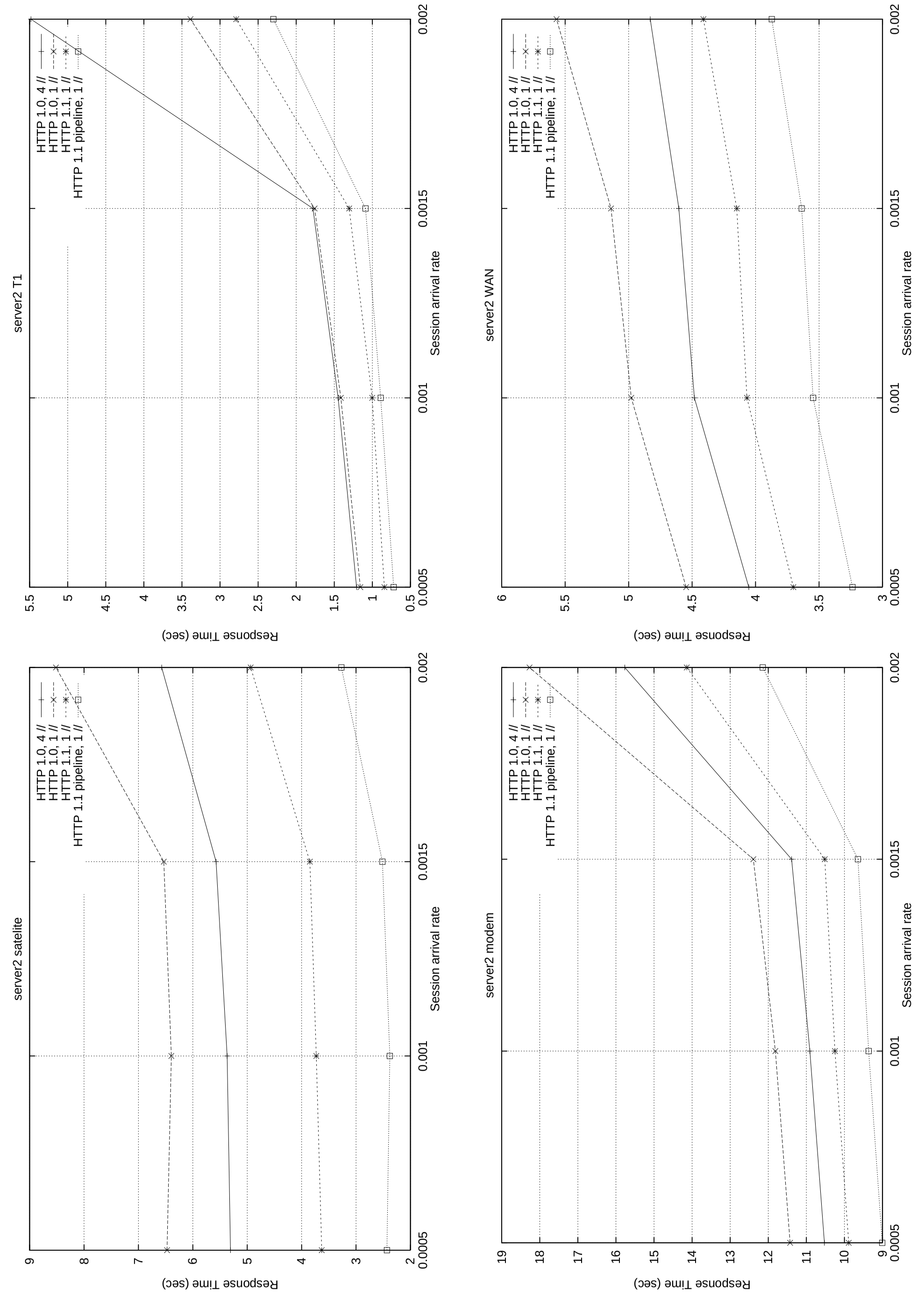

Fig. 4. Performance comparison between HTTP1.0 and HTTP1.1 on server 2

and this is true even if the latter uses four parallel connections while the former just one connection. In most cases, HTTP1.1 with pipeline outperforms HTTP1.1 without pipeline, and HTTP1.1 without pipeline outperforms HTTP1.0 with four parallel connections, and the latter outperforms HTTP1.0 with a single connection. The difference in request response time becomes more significant when the network is heavily loaded. The response time of clients using HTTP1.0 with a single connection could be ten times larger than those using HTTP1.1.

These results are coherent with those of [7]. In our experiments, the number of packets generated by HTTP1.0 (103101 in one of the experiments) is almost twice of that by HTTP1.1 (58896 for the same experiment without pipeline and 57617 with pipeline). This saving in the number of packets by HTTP1.1 explains the significant difference in response times when the network is heavily loaded.

References

[1] Paul Barford and Mark E. Crovella. Generating representative web workloads fot network and server performance evaluation. In *Proceedings of ACM Sigmetrics'98*, 1998.

[2] Mark E. Crovella, Murrad S. Taqqu, and Azer Bestavros. Heavy-tailed probability distributions in the world wide web. To appear in the book: A Practical Guide To Heavy Tails: Statistical Techniques for Analysing Heavy Tailed Distibutions., 1996.

[3] N. Likhanov, B. Tsybakov, and N.D. Georganas. Analysis of an atm buffer with self-similar input traffic. In *Proceedings of Infocom'95*, April 1995.

[4] Binzhang Liu. Characterizing web response time. Master's thesis, Virginia Polytechnic Institute and State University, April 1998.

[5] S. Manley, M. Courage, and M. Seltzer. A self-scaling and self-configuring benchmark for web servers. Technical report, Harvard University, 1998.

[6] David Mosberger and Tai Jin. httperf–a tool for measuring web server performance. In *Workshop on Internet Server Performance (WISP'98)*, Madison, WI, June 1998.

[7] H. Frystyk Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, H. Lie, and C. Lilley. Network performance effects of http/1.1, css1, and png. In *Proceedings of the ACM SIGCOMM '97*, Cannes, France, September 1997.

[8] V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. In *Proceedings of ACM/Sigcomm'94*, pages 257–268, September 1994.

[9] SPEC. An explanation of the specweb96 benchmark, 1996.

[10] Gene Trent and Mark Sake. Webstone: The first generation in http server benchmarking, 1995.