How Well Does File Size Predict Wide-Area Transfer Time?

Manish Sharma John W. Byers
Department of Computer Science, Boston University
Boston, MA 02215, USA

***Abstract*—**
In scheduling connections at busy web servers, it is commonly assumed that transmission duration (or time in system) is directly proportional to the size of the file transferred. For example, a scheduling discipline such as SRPT (shortest remaining processing time) could use this assumption to order connections according to the residual size of the transfer. However, with a diverse client population, network effects such as packet loss, heterogeneous end-to-end bandwidths and latencies render this assumption invalid. In this measurement study, we explore this relationship and investigate the predictive value of file size in determining transfer time. We use the publicly available sanitized cache access logs which are collected on a daily basis as a part of IRCache [16], the NLANR Web caching project, to explore this relationship for HTTP traffic serviced by the NLANR caches over a weeklong interval. Over this dataset, we first confirm an earlier finding: that for small transfers of up to 30KB, there is virtually no correlation between file size and transfer time; moreover, transfer times vary over 5 orders of magnitude. For larger files, we find that file size and transfer time are increasingly well correlated as file size increases, but we still find that predictions of transfer time from file size alone are not highly accurate. Our findings motivate further investigation of incorporating network-awareness into end-system scheduling disciplines.

I. Introduction

In network applications ranging from connection scheduling to Web caching to admission control, high performance is typically contingent on accurate predictions about the future state of the system. But, building accurate predictive models is challenging, as predictions are necessarily made with incomplete information. Network conditions are highly variable, and policies must cater to a heterogeneous mix of clients and connections. But accurate models are difficult to obtain and for this reason, simple models of network conditions are often used (either implicitly or explicitly) in practice.

One such model is used in the context of scheduling connections at busy webservers. A sequence of modelling and analysis work has demonstrated the effectiveness of using the SRPT (shortest remaining processing time) scheduling discipline [5], [3]. But determining *which* connection in fact has the shortest remaining time in system is a difficult prediction problem that has not been the main focus of prior work. The authors advocating SRPT use the size of the file, or more precisely, the number of bytes remaining to be transferred, as an estimate of the time this job will remain in the system. While larger files are clearly indicative of longer transfers *in general*, exclusive use of this heuristic may provide poor predictions for specific connections, as it does not account for diversity of network conditions across a set of heterogeneous clients. In this paper, we investigate the relationship between file size and transfer duration and study the predictive value of file size using measurement data collected from Internet logs comprising millions of HTTP transfers. We view the study of this relationship as fundamental, since on the one hand, if such a simple measure has strong predictive value, it should be championed; while if its shortcomings are serious, the need for better methods is apparent.

While there are several studies which consider some aspect of the relationship between file size and transfer duration (we survey them in Section II), to our knowledge and to our surprise, there is no definitive study of this relationship using recent Internet measurements. The starting point of our measurement study is the daily Web logs collected by proxy caches administered by NLANR as part of the IRCache project. We describe how these logs were collected, what data they record and the methodology we use to extract meaningful file size and transfer time estimates from the logs in Section III. We then break down a dataset from a representative one week interval at one of the busiest of these proxies and provide a distilled summary of our findings in Section IV. Our paper concludes with a brief recap and a sketch of our ongoing work in Section V.

Email:{manishrs, byers}@cs.bu.edu
This work supported in part by NSF grant ANI-9986397 and by NSF CAREER Award ANI-0093296.

II. Related Work

File size has long been used as a metric in a wide range of policies and algorithms used to efficiently deliver WWW content. Irani was first to observe that caching policies which do not take file size into account do not work well in practice [9]. Modern streaming media caching algorithms such as Resource-Based Caching consider both the impact of file size and required delivery bandwidth in making cache insertion and replacement decisions [15]. Size-based scheduling policies at Web servers use the size of the requested file to prioritize connections which request smaller files [3], [8]. Most of these works assume that transmission time primarily depends on the size of the requested file. For instance, [3], [8] use file size as a measure of the size of the connection then use it to optimize their scheduler so as to improve the "client experience," or the time taken by the client to receive the requested file. However, they do not study the question of whether the size of the requested file is the only measure or the best measure for improving performance of their SRPT scheduler.

On the measurement and characterization side, there has been substantial work done to measure and understand both the latencies incurred in WWW transfers and the performance at the server. Workload characterization studies such as [1] consider the distribution of file size requests and request interarrival times, but do not record the elapsed time for document transfers. Barford and Crovella measured the latencies of fixed file sizes over combinations of server load, network load and file sizes [4]. While this study considers the impact of file size, server load and network conditions on transfer time, the client population they studied was both small and fairly homogeneous, moreover they did not aim at characterizing the correlation between file size and transfer time.

Finally, there are a handful of measurement studies which do consider the relationship between file size and transfer time, but these studies are both dated and limited in their scope. This may be in part due to the dearth of publicly available data which records the transmission durations of files of various sizes to a set of heterogeneous clients from the perspective of a single server. One example of such a study is [2] where the authors derived conclusions about TCP behavior of a busy internet server. They analyzed the traces collected at the official web server of the 1996 Atlanta Olympic games. Such traces could be used to study the relationship between file size and transmission time but they are not publicly available. The work closest to ours is the work of Mogul [12], who analyzed the logs of a busy web server which provided live election results of the 1994 CA Elections over a 24-hour period. They show that there is essentially no correlation between retrieval size and elapsed time for sizes below about 30KB, since overhead costs of creating TCP connections, creating server processes, exchanging and parsing request headers and sending response headers overwhelm any per-byte costs for smaller files. However, this work does not study the correlation between file size and transfer time for bigger files. Cunha et al [7] demonstrated correlation between transfer time and file size, but their experiments were performed over a limited file set (a total of approximately 600,000 requests over a 3 month period) and across a population of primarily local, homogeneous clients. Finally, in [11], Liu and Fox studied the correlation between file size and transmission duration from the perspective of a client, in a fairly restricted experimental environment. As with other datasets, their client base is homogeneous, the range of requested file sizes was limited, and the transfers do not all originate at a single, fixed server. In our work, we analyze logs collected at proxy servers which serve a wide client base. The requests served by the proxy server originate at different locations and the client base provides a heterogeneous environment with respect to the type of requests as well as the last-hop technologies used by hosts which make HTTP requests.

III. Data Collection and Methodology

In order to relate transmission durations to file sizes, we need server-side logs which record both the transmission time and the file size. Moreover, we desire that the transfers be diverse, both in terms of the sizes of the files transferred and in terms of the clients who access them. Even logs which address the first point are difficult to find, since few traces exist which record the transfer times of files. Most of the publicly available logs which are collected at servers or proxy servers simply record the timestamp when the first byte of the requested file is transferred, and generally do not record the time when the last byte of the requested file is transferred. Another challenge is that organizations which operate busy servers which service a highly diverse set of clients often do maintain detailed logs, but are reluctant to share their data with researchers, for reasons ranging from competitive factors to privacy issues.

The one set of logs we identified as suitable for our study is the publicly available sanitized cache access logs which are collected on a daily basis as a part of IRCache [16] which is the NLANR Web caching project[1]. While IRCaches are not servers, strictly speaking, they serve as endpoints of a diverse set of wide-area connections across which transfer times and transfer sizes can be measured accurately. The IRCache mesh consists of 10 medium/large caches in the United States. More information on the IRCache server names and their locations can be found in the FAQ section of [16].

On average an IRCache server accepts 5 to 40 connections per second and the Median Object Size, i.e, the number of bytes written to the client socket, is 3600 bytes. The hardware platform is either Digital Alphaserver 1000's (circa 1995) with 512 MB RAM, and about 24 GB disk space, or custom-built Intel-based PCs with about the same amount of RAM and disk. Digital boxes use Digital Unix as the operating system and the PC's run FreeBSD. All caches run Squid [14], which was also supported by the IRCache project. Squid is a high-performance proxy caching server for web clients, supporting FTP, gopher, and HTTP data objects. Squid works by caching internet objects which are files requested by users. A client requests an internet object from a caching proxy; if the object is not already cached, the proxy server fetches the object (either from the host specified in the URL or from a parent or sibling cache) and delivers it to the client. The Internet Cache Protocol (ICP) is used for communication among squid caches, primarily within a cache hierarchy to locate specific objects in sibling caches. If a squid cache does not have a requested document, it sends an ICP query to its siblings, and the siblings respond with ICP replies indicating a "HIT" or a "MISS." The cache then uses the replies to choose from which cache to resolve its own MISS.

As a part of the IRCache project, Squid maintains daily logs which are sanitized and made available to researchers on demand. The logs contain information such as request duration, some timeout information, the next upstream server address, and the content type etc. The fields which are of interest to us are:

1. Duration: The elapsed time reports how many milliseconds the transaction busied the cache. For HTTP/1.0, this the time between accept() and close().

2. Bytes: The total number of bytes delivered to the client including the headers. Also, failed requests may deliver a short error page, the size of which is also included.

3. Result Codes: This column encodes the transaction result, including the cache result, which contains information on the kind of request, how it was satisfied, or in what way it failed.

The result codes in particular are important to our methodology, thus we explain the important ones here. Interested readers may refer to Squid FAQ [14] for other codes. The `TCP_` codes refer to requests on the HTTP port (usually 3128).

`TCP_HIT`: A valid copy of the requested object was in the cache.

`TCP_MEM_HIT` : A valid copy of the requested object was in the cache and it was in memory, thus avoiding disk accesses.

`TCP_IMS_HIT` : The client issued an If Modified Since request for an object which was in the cache and fresh.

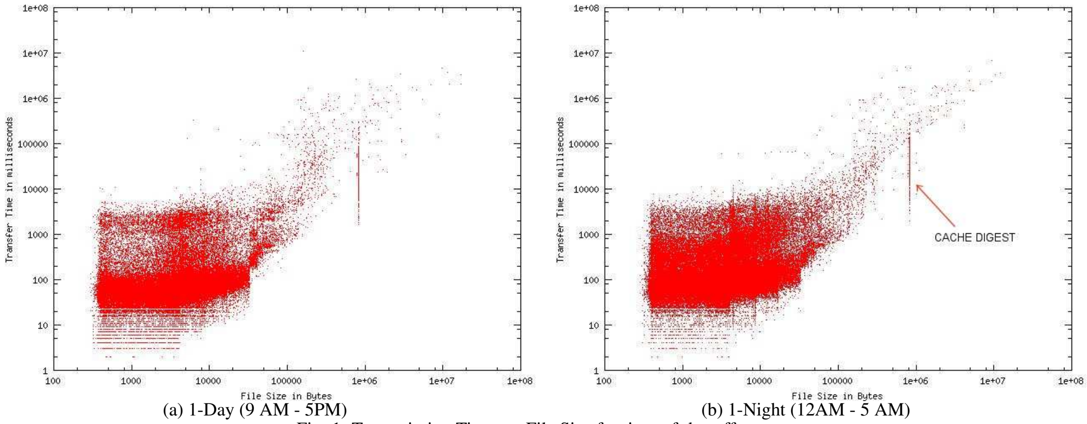
The `TCP_` codes which end with a `MISS` represent requests for which the object was not found in the cache. Since a `MISS` results in fetching an object from a different cache/host, we consider only those records in the logs which have `TCP_HIT`, `TCP_MEM_HIT` or `TCP_IMS_HIT` as result codes. We also note that Squid's measure of transfer time does not account for the time required for connection setup.

We extract the Duration and the Bytes field from this data. Comparative plots of file size and transmission duration using these fields are the main focus of our study in the next section.

A. Caveats

Before we present our results, we precede our findings with some caveats. For reasons of space, we present a study of a

[1] IRCache is the NLANR Web Caching project, originally funded by the National Science Foundation, Directorate for Computer and Information Sciences and Engineering. From 1996-2000, the IRCache project was administered by the University of California San Diego, in cooperation with the San Diego Supercomputer Center and the National Laboratory for Applied Network Research.



(a) 1-Day (9 AM - 5PM) (b) 1-Night (12AM - 5 AM)

Fig. 1. Transmission Time vs. File Size for time-of-day effects

single server within the IRCache mesh for a weeklong interval. While we have considered many other logs, we do not present them here, as we believe that this dataset provides a representative reflection of IRCache activity. While these effects are not insignificant, we believe they do not significantly modify the findings presented in this paper. Also, pruning out the `MISS`es from the proxy logs results in transfers of unpopular files being underrepresented in our dataset, a factor which could potentially impact the results of our measurements.

Finally, we do not have similar datasets from sources other than IRCache traces to validate our findings – older datasets used in previous work described in Section II offer a much less diverse collection of transfer times and file sizes and do not provide a meaningful point of comparison. On the other hand, the IRCache logs by themselves provide a very rich collection of wide-area transfers as a reference study.

IV. Results and Observations

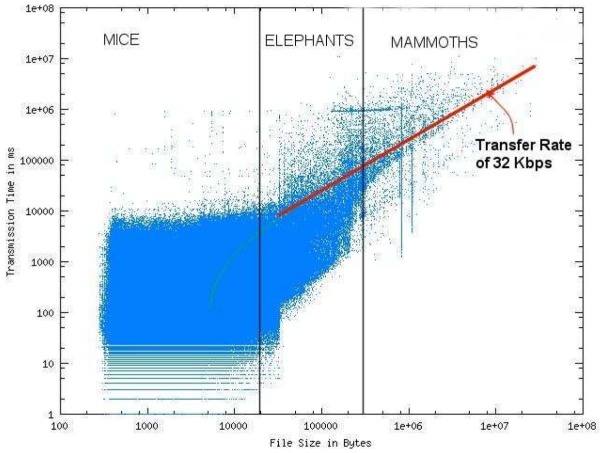


Fig. 2. Transmission Time vs. File Size for a 5-day aggregate period

We studied trace files collected from ten cache sites over the period of several months. To provide a representative snapshot of the request distribution at a single server (other logs were similar), we studied the trace files collected over a weeklong period from February 2, 2002 to February 8, 2002 from a cache site at Digital Internet Exchange in Palo Alto, California. In Figs. 1(a) and (b), we depict transfers on the day and night of Feb. 5, a representative weekday in the weeklong period. In each case, after pruning out the `MISS`es as described in Section III, we plotted file size vs. transmission time for each transfer on a log-log scale. Since transfers during weekdays and weeknights of this period were quite similar, we aggregated the data for this 5 day interval into our main dataset, shown in Fig. 2. This dataset consists of approximately 1.3 million transfers with file sizes varying from 280 B to 28 MB with a mean of 16KB and a median of 4.5KB.

For the purpose of exposition, we divided files into three file size categories reflecting visually distinct regions of Fig. 2 as described in Table I. As a point of reference, the line on Fig. 2 corresponds to a transfer rate of 32 Kbps. In all of the traces, there are certain sets of anomalous transfers. One such is labeled "Cache Digest" in Fig. 1(b) and we discuss this further in Section IV(C). A statistical summary of the dataset is also shown in Table I. Due to lack of space we are not able to present characteristics of file sizes on a finer granularity than across these three regimes. As expected the majority of transfers are for files up to approximately 50KB, as dictated by the heavy-tailed nature of WWW file sizes [6]. As the file size increases, there is a gradual decrease in the density of points as there are relatively few big file requests. Mice constitute 94.4% of the total transfers, elephants constitute 5% and mammoths constitute 0.6% of the total set of file transfers. time

A. Are File Sizes and Transfer Times Correlated?

From the plot in Fig. 2, there is a marked trend toward correlation between file size and transfer time as the file size grows large, so it is clear that at least some degree of correlation exists. To confirm this observation, we performed statistical tests to measure the degree of correlation between our random variables. One standard method is to compute the Pearson Product Moment Correlation. The formula for calculating the correlation takes on many forms – one that is commonly used [10] is:

$$r = \frac{\Sigma XY - \frac{\Sigma X \Sigma Y}{N}}{\sqrt{(\Sigma X^2 - \frac{(\Sigma X^2)}{N})(\Sigma Y^2 - \frac{(\Sigma Y^2)}{N})}} \tag{1}$$

where X and Y are the random variables which are being correlated, N is the number of trials and r is the correlation coefficient. Table I shows the correlation for the groups of transfers classified as mice, elephants and mammoths. Using this measure, a correlation of greater than 0.01 is considered significant. Clearly, transfers classified as mice show very weak correlation between file size and transfer time, whereas mammoths show a strong correlation.

TABLE I

SUMMARY OF THE DATASET

Category	Size Range	Number of Transfers	Mean File Size	Median File Size	Mean Transfer Time	Median Transfer Time	Correlation
Mice	Below 30 KB	1251056	6.1 KB	4.2 KB	501 ms	129 ms	0.016
Elephants	30 KB - 500 KB	66146	75 KB	48 KB	20800 ms	1350 ms	0.337
Mammoths	Above 500 KB	8217	1.1 MB	0.83 MB	243 s	46.1 s	0.538

The weak correlation between file size and transfer time for small files (also observed in [12]) can be attributed to various factors, network conditions being among the most important. The short transfers classified as mice in our dataset typically consist of only a handful of packets, thus they may spend all of their time in TCP slow-start, where network latency plays a more dominant role than available network bandwidth in determining throughput. Related work [2] has also shown that when a single bad event occurs in a short TCP transfer, such as a packet loss or a lost FIN packet, the transfer time may increase by an order of magnitude or more due to TCP's poor loss recovery performance for short transfers. For these reasons, we argue that file size alone cannot provide meaningful predictions as to the duration of a small transfer over the wide-area.

B. Correlation vs. Predictiveness

For large transfers, the results provide a different perspective. In contrast to the striking lack of correlation seen for small transfers, file sizes and transfer times become increasingly well correlated as file size increases. This is to be expected, as the transfer time is primarily dominated by bandwidth rather than latency, and the throughput of the transfer can be accurately modeled by standard equations expressing TCP throughput as a function of network conditions along the path [13]. Therefore two clients experiencing identical network conditions should receive roughly the same throughput, with transfer time linear in the sizes of the files they are downloading. Indeed, if all network paths were created equal, we would expect to see points clustered along a line $y = Ax + c$, where A is the throughput along that path in bytes, in which case the predictive power of file size in determining transfer time would be ideal.

But this is not the case, i.e. for any given file size, transfer times range over several orders of magnitude. Thus it is clear that correlation does not immediately imply strong predictive power. In other words, if we try to derive a predictor based on file size to predict transfer duration, we see that it will often be inaccurate, and predict the transfer duration poorly. To formalize this notion of predictiveness, we use the following definition.

Definition 1: Let $\{D_i\}$ be a family of input distributions over the reals indexed by the positive integers i and let r_x denote a random variable drawn from the distribution D_x. We say that a predictor function $f(x)$ has (δ, ϵ)-predictiveness for the family $\{D_i\}$ if for each individual x,

$$Pr[f(x) - \delta \leq r_x \leq f(x) + \delta)] \geq 1 - \epsilon.$$

Applying this definition to our problem domain, we think of D_x as being the distribution of measured transfer times for files of size x. We let $f(x)$ be a predictor for the transfer time – one reasonable candidate function is the linear predictor mentioned earlier, another is a non-linear predictor which provides the minimum-error fit to the measured distributions $\{D_i\}$. The parameter δ is the distortion parameter, or a measure of the amount of deviation we allow in the actual transfers to warrant calling a prediction a success. Finally, ϵ is the failure probability, or the probability that the actual transfer time lies outside the amount allowed by the distortion parameter. The above definition implies that a good predictor function should predict the transmission durations with a probability of $(1 - \epsilon)$ with accuracy of $\pm\delta$ for small values of ϵ and δ.

To quantify the effectiveness of transfer time prediction based on file size, we computed the δ needed to achieve different accuracies ϵ for different file sizes. For example, to compute δ for $\epsilon = 0.2$ for a given (narrow) range of file sizes, we consider all the transfer times for this size, and take the 10th percentile and the 90th percentile transfer time, denoted by t_1 and t_2, then take $\delta = (t_2 - t_1)/2$.

Fig. 3(a) shows the δ-curves for a representative set of file sizes for different accuracy levels given by $\epsilon = 0.2, 0.1, 0.05$. It can be clearly seen from these curves, that for any given file size, the value of δ is high. Thus, even when outlying transfers are discarded, mainstream transfer times still span several orders of magnitude for all file sizes. Of course, to achieve smaller values of ϵ, larger values of δ are needed across all file sizes. The most accurate predictions occur when δ is smallest, thus transfers in the range of 1KB - 10KB appear to be the most predictable.

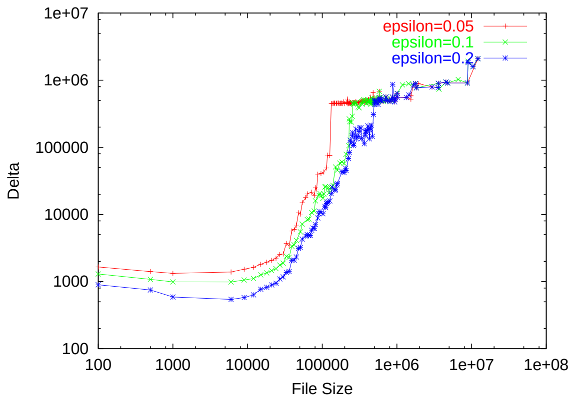
In conclusion, for file transfers of all sizes, it is clear that the variation in transfer times is too large to enable a predictor function to predict transmission durations accurately based on file size alone.

C. Cache Communication

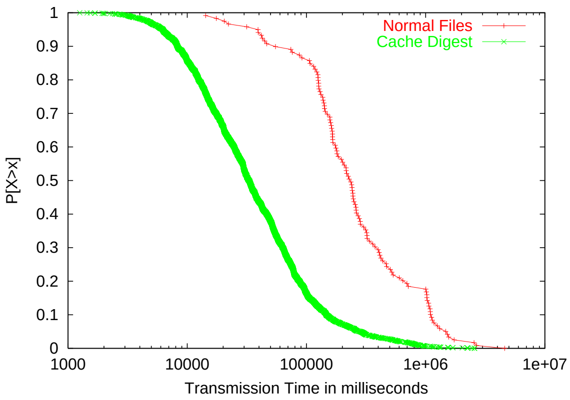
There are certain files amongst the big files which exhibit a different behavior with respect to their transmission durations. These files are the distinct vertical lines marked on Fig. 1(b) and also visible in Fig. 2; they are accessed far more times than the other files of the same magnitude. On close examination, we found that these files are cache digests which are exchanged between different proxies in the mesh from time to time. They are cache summaries which are Bloom Filters representing the contents (URLs) of the cache. The request is made as an HTTP request and hence the transfer is captured as a normal HTTP transfer. Neighbors request the digest to lookup URLs, as an alternative to ICP. We present the CCDF of the transmission times of these HTTP transfers in Fig. 3(b). We also present the CCDF of the transmission times of a set of normal HTTP transfers of files of approximately the same size (0.8 MB) for comparison. The cache digest file was accessed 3701 times during the 5-day measurement period. The CCDF for a normal HTTP transfer shows less variation in the transmission duration as compared to the CCDF of the cache digest which shows that transfer times have a wider range.

D. Unexplained Phenomena

The plot in Fig. 2 shows a region of transfers which connect the mice and the mammoths. We denote the transfers



(a) Delta vs. file size for different ϵ



(b) CCDF of a cache digest transfer vs. a normal file transfer

Fig. 3. δ-Curves and CCDFs of Cache digests vs. Normal files

in this region as elephants and it corresponds to file sizes between 30 KB and 500 KB. The interesting observation here is that there is a clear and distinct boundary between mice and elephants where the transmission time and file size relationship undergoes an abrupt change. Intuitively, an increasing correlation between file size and transfer time with increasing file size might suggest a more gradual transition from mice to elephants with respect to transmission time. We do not have an explanation for this behavior but it provides an interesting area of study.

V. Conclusion

It is clear from our study that transfer time prediction based purely on file size has serious shortcomings especially under diverse network conditions and in the presence of a heterogeneous set of clients. While small file transfers are latency driven and show a weak correlation between file size and transfer time, big transfers are bandwidth driven and show an increasing correlation as the file size increases. Transfer time for smaller files cannot be predicted meaningfully from the size of the requested file. Moreover, even though there is a better correlation between file size and transfer time for large files, correlation does not imply a strong degree of predictiveness. In practice, transfer times vary over several orders of magnitude irrespective of the file size, and predicting transfer times based on file size alone is necessarily inaccurate.

Our future work focuses on network applications and policies which can stand to benefit from a better understanding of the relationship between file size and transfer time, and from better accounting for network conditions while estimating transmission durations of file transfers. A potentially interesting direction is that of predicting transfer times of objects other than static files, such as dynamic content, or rich multimedia content. For example, with an increasing shift from static HTTP requests to cgi-scripts and other non-static requests, accurate prediction becomes more complex. Similarly, understanding the transmission durations of rich multimedia streams, which typically contain embedded objects, could be useful in predicting the lifetimes of multimedia sessions and in allocating web resources accordingly.

Acknowledgements

We would like to thank Duane Wessels for giving us access to the IRCache traces and for providing us with useful information about the Squid Proxy Server. We would also like to acknowledge helpful conversations with Mark Crovella.

References

[1] M. Arlitt and C. Williamson. Internet Web Servers: Workload Characterization and Performance Implications. In *IEEE/ACM Transactions on Networking*, October 1997.

[2] H. Balakrishnan, V. Padmanabhan, S. Seshan, M. Stemm, and R. Katz. TCP Behavior of a Busy Internet Server: Analysis and Improvements. In *Proceedings of INFOCOM*, pages 252–262, 1998.

[3] N. Bansal and M. Harchol-Balter. Analysis of SRPT Scheduling: Investigating Unfairness. In *ACM SIGMETRICS*, June 2001.

[4] P. Barford and M. Crovella. Measuring Web Performance in the Wide Area. In *Performance Evaluation Review*, September 1999.

[5] M. Crovella, R. Frangioso, and M. Harchol-Balter. Connection Scheduling in Web Servers. In *USENIX Symposium on Internet Technologies and Systems*, June 1999.

[6] M. Crovella, M. Taqqu, and A. Bestavros. Heavy-tailed probability distributions in the world wide web, 1998. In A Practical Guide To Heavy Tails, chapter 1, Chapman and Hall, New York, pp. 3–26.

[7] C. Cunha, A. Bestavros, and M. Crovella. Characteristics of WWW Client-based Traces. Technical Report 1995-010, Boston University, January 1995.

[8] M. Harchol-Balter, N. Bansal, and B. Schroeder. Implementation of SRPT Scheduling in Web Servers. Technical Report CMU-CS-00-170, Carnegie Mellon University, 2000.

[9] S. Irani. Page Replacement with Multi-size Pages and Applications to Web Caching. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, May 1997.

[10] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley and Sons Inc., 1991.

[11] B. Liu and E. Fox. Web traffic latency: Characteristics and implications. In *WebNet98*, November 1998.

[12] J. Mogul. Network Behavior of a Busy Web Server and its Clients. Technical Report 95.5, Compaq Western Research Lab, October 1995. http://research.compaq.com/wrl/techreports/abstracts/95.5.html.

[13] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *Transactions on Networking*, pages 303–314, April 2000. A preliminary version of the paper also appeared in ACM SIGCOMM 1998.

[14] SQUID Webpage. http://www.squid-cache.org.

[15] R. Tewari, H. Vin, A. Dan, and D. Sitaram. Resource-Based Caching for Web Servers. In *Proceedings of SPIE/ACM Conference on Multimedia Computing*, January 1998.

[16] The IRCACHE Project. http://www.ircache.net.