Time-Domain Analysis of Web Cache Filter Effects (Extended Version)

Guangwei Bai, Carey Williamson

Department of Computer Science, University of Calgary, 2500 University Drive NW, Calgary, AB, Canada T2N 1N4

Abstract

This paper uses trace-driven simulation to study the traffic arrival process for Web workloads in a simple Web proxy caching hierarchy. Both empirical and synthetic Web proxy workloads are used in the study.

The simulation results show that a Web cache reduces both the peak and the mean request arrival rate for Web traffic workloads, while the variance-to-mean ratio of the filtered traffic typically increases, depending on the input arrival process and the configuration of the cache. If the input traffic is self-similar, then the filtered request traffic remains self-similar, with the same Hurst parameter, though with reduced mean. Finally, we find that a Gamma distribution provides a flexible and robust means of modeling aggregate workloads in hierarchical Web caching architectures, for a broad range of workload characteristics and Web proxy cache sizes. To demonstrate the generality and effectiveness of the modeling approach, we present a detailed example of filter effects and traffic superposition in a two-level Web caching hierarchy with heterogenous input workloads. The Gamma modeling results match well with the results from trace-driven simulations.

Key words: Internet and WWW Technology, Web Proxy Caching, Web Traffic Simulation, Workload and Traffic Characterization

1 Introduction

The World Wide Web (WWW, or the Web) continues to be a major driving force behind the growth in popularity of the Internet. The Web has become

Email address: {bai,carey}@cpsc.ucalgary.ca (Guangwei Bai, Carey Williamson).

the preferred means for the timely dissemination of information in research, education, news, marketing, travel, business, and entertainment domains.

The explosive growth of the Web, with its corresponding increase in Internet traffic volume, has led to user-perceivable network performance problems. In some cases, the performance bottlenecks are at the Web server, if the server architecture cannot handle the client demand. In other cases, the bottleneck is within the Internet itself; network congestion leads to queueing delays and packet losses that degrade Web performance. In yet other cases, excessive delays are due to inefficiencies in the Internet protocol stack (e.g., the interactions between HTTP and TCP), round trip delays across the Internet, or limited client access bandwidth to the Internet.

Web document caching (Web caching) architectures and content distribution networks (CDNs) are now widely used to alleviate these performance problems. By storing copies of popular Web documents close to the users requesting them, Web caches can reduce Web server load and can reduce the volume of traffic traversing the core of the Internet. In many cases, users perceive improved response times for document downloads.

To enhance the performance of Web caching, multi-level Web caching hierarchies have recently received increasing research attention [10,16,21,29]. An effectively-organized multi-level Web proxy caching hierarchy can improve Web performance, for instance, by using different approaches to cache management at each level of the hierarchy [12,29].

The presence of a cache has a *filter effect* on Web workloads. Because some of the incoming client requests are satisfied as hits at the Web proxy cache, these requests are removed (filtered) from the request workload progressing upstream to the Internet, to other caches, or to the origin servers. This filtering effect is illustrated conceptually in Figure 1, where the original aggregate client workload from an organization traverses Link1 to the organization's Web proxy cache, and the filtered workload traverses Link2 en route to the Internet. The filtering effect in turn changes the response traffic volume traversing Link2 en route to the clients.

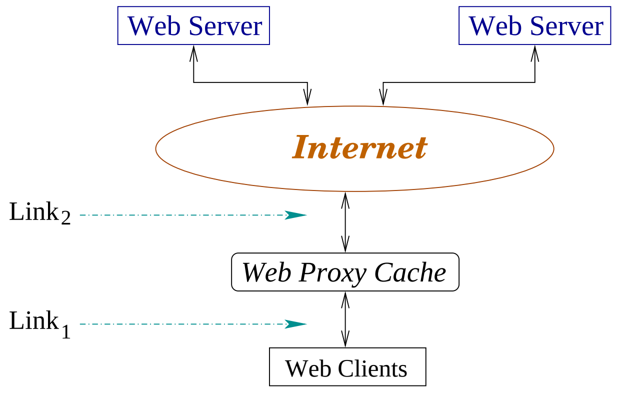


Fig. 1. Conceptual Illustration of Web Cache Filtering Effect

The cache filtering effect manifests itself in two orthogonal ways. First, the cache generally filters out requests for the most popular Web documents (depending of course, on the cache management policy). As a result, the number of HTTP requests for certain Web documents is substantially reduced by the presence of the cache [29]. Second, the presence of the cache changes the structure of the request arrival process entering the Internet, compared to the request arrival process presented to the cache itself.

The first of these two effects has been fairly well studied in the literature [15,28,29]. We refer to this work as *frequency-domain* analysis of cache filter effects, since it focuses largely on the frequency distribution of Web document popularity. This document popularity profile typically has a power-law structure before the cache, often characterized using a Zipf or a Zipf-like distribution [8,9,21]. After the cache, the "high popularity" end of the Zipf distribution is significantly flattened [15,29]. The precise manifestation of the cache filter effect is highly dependent upon the architecture of the Web proxy caching system, the cache size used, the cache replacement policy, and Web workload characteristics [10,29].

Relatively little research work has focused on the second aspect of cache filter effects: the impact on the request arrival process. Clearly, the presence of the cache reduces the average arrival rate of requests entering the Internet. Indeed, this is the primary motivation for many organizations to install a proxy cache in the first place. In most cases, the cache reduces the peak arrival rate as well, though not necessarily in the same proportion as the mean. However, the impact on the variability (burstiness) and self-similarity [14] of the traffic is not clear.

The purpose of this paper is to study cache filter effects on the traffic arrival process. We refer to our work as *time-domain* analysis of cache filter effects, to distinguish it from the former frequency-domain effect. Our research is carried out using trace-driven simulations, with empirical and synthetic Web proxy workloads, along with tools for Web caching simulation and traffic characterization and modeling.

The research questions addressed in this paper are:

- What impact does a Web proxy cache have on the structural characteristics (i.e., mean, peak, variance, self-similarity) of a Web request workload?
- How sensitive is the filter effect to the cache size and the cache replacement policy used?
- How sensitive is the filter effect to the characteristics of the incoming Web workload (i.e., Zipf slope, self-similarity)?
- Can a closed-form mathematical model adequately characterize the cache filter effect?
- How can we model the aggregate Web request streams in a multi-level Web proxy caching hierarchy?

We address these questions using trace-driven simulations of a multi-level Web proxy caching hierarchy. The simulation experiments quantify the filter effects of a Web cache on the request arrival process, for both empirical and synthetically-generated aggregate Web client workloads. Our results show that a Web cache reduces both the mean and the peak arrival rate for Web traffic workloads. The presence of the cache seems to have no impact on the degree of self-similarity in a workload, though in typical cases the filtering effect reduces the variance of the outbound request stream. The study also demonstrates that the superposition of Web request streams from multiple child caches in a Web proxy caching hierarchy does not result in smooth traffic. Rather, multiplexing bursty request streams tends to produce a bursty aggregate stream. Finally, we find that a Gamma distribution provides a simple, flexibile, and relatively robust means of characterizing the request arrival count process, both before and after a Web cache. The parameters for fitting the Gamma distribution can be estimated from empirical/synthetic traffic traces, though the fitted parameters for describing the filtered request stream are strongly dependent on the cache size and the characteristics of the input Web workload. The choice of cache replacement policy has relatively little impact on the traffic structure.

To demonstrate the generality and effectiveness of the Gamma modeling approach, we present a detailed example of filter effects and traffic superposition in a two-level Web caching hierarchy. We consider heterogenous input workloads, and analyze the transformation of the workloads proceeding through the caches. The Gamma modeling results for the output request streams agree closely with the trace-driven simulation results for the same workloads. To the best of our knowledge, these are the first modeling results characterizing workloads throughout a Web caching hierarchy.

The remainder of this paper is organized as follows. Section 2 discusses related work on Web workload characterization and cache filter effects. Section 3 describes the experimental methodology for our study, as well as the empirical and synthetic Web proxy workloads used. Section 4 presents simulation results for Web cache filter effects on the request arrival process, using an empirical Web workload. Section 5 presents additional simulation results for synthetic Web workloads. Section 6 proposes and validates a parameterizable model for characterizing Web request streams, both before and after a cache. Section 7 focuses on the superposition of self-similar Web workload streams in a multi-level caching hierarchy, and the modeling of aggregate Web workload in the time-domain. Finally, Section 8 concludes the paper.

2 Background and Related Work

2.1 Web Proxy Caching

Web proxy caching is a technique used for improving Web performance on the Internet. Web proxy caches are located between Web clients (browsers) and Web servers. Proxies accept client requests and forward them to Web servers only as necessary. When a requested document is returned by a Web server, the proxy sends the document to the client and stores a copy of the document in its local cache, in the hope that the proxy can satisfy future client requests for the same document without contacting the origin server, thus reducing user-perceived response time.

Caching effectiveness is traditionally measured by two quantities: the (document) *hit ratio* is the percentage of the total requests that are satisfied directly by documents stored in the cache; and the *byte hit ratio* is the percentage of the total requested Web content bytes that are satisfied directly by documents stored in the cache. Both metrics are required since Web objects vary significantly in size. Other metrics such as user-perceived response time are dependent upon the hit ratio and the byte hit ratio, as well as other factors such as the network bandwidth, the round-trip delay, server load, and network congestion [23].

Multi-level Web proxy caching systems are used to improve the performance and scalability of the Internet. In a hierarchical configuration, proxies at or near the end-user constitute the lowest level of the hierarchy, often with sibling-sibling relationships with one another. The lowest level proxies may have a child-parent relationship to a higher level proxy, usually a (geographically) regional proxy [29]. A regional proxy can in turn connect to a higher level proxy, such as a national proxy [21]. A request that cannot be satisfied by one proxy cache can be sent to a nearby sibling or to the parent using an

Inter-Cache Protocol. Contacting the origin server to obtain the document is the last resort [29].

2.2 Web Workload Characterization

Several Web workload characterization studies have appeared in the literature. These studies have focussed on Web client [7], Web server [3], and Web proxy workload characteristics [2,21].

These empirical studies identify several common workload characteristics that are relevant to Web caching performance. These characteristics include a high degree of *one-time referencing*, a *Zipf-like document popularity distribution*, *heavy-tailed file and transfer size distributions*, and a *temporal locality property* in the document referencing behaviour [2,3,7,21]. These characteristics are quite well-documented in the literature, and are thus not discussed at length here.

Among these characteristics, the one that is most relevant to Web caching performance is the slope of the Zipf-like document popularity distribution [9]. Zipf's law expresses a power-law relationship between the popularity P of an item (i.e., its frequency of reference) and its rank r (i.e., relative rank among the referenced items, based on frequency of reference). This relationship is of the form $P = c/r^\beta$, where c is a constant, and β is often close to 1. For example, the frequency of usage for English words in written prose typically follows this distribution.

In the Web context, a similar referencing behaviour is observed [8,21,27]. Some researchers have found that the value of the exponent β is close to 1 [2,7], precisely following Zipf's law. Others [2,8,21] have found that the value of β is less than 1, and that the distribution can be described only as "Zipf-like", with the value of β varying from trace to trace. In general, the steeper the Zipf slope, the higher the cache hit ratio achievable for a given Web workload [8,9,27].

2.3 Web Traffic Self-Similarity

Recent research in network traffic measurement has challenged the Poisson assumptions in traditional network traffic models [1,13,20,25]. Leland *et al.* [20] showed that Ethernet traffic is bursty across many time scales, and can be described statistically as *self-similar*. Crovella *et al.* [13] observe self-similarity in Web traffic as well. The term self-similar means that the statistical characterization of the traffic is essentially invariant with time scale; the same statistical properties are observed at time scales of milliseconds, seconds, minutes, hours, and more. Self-similarity often arises from the presence of heavy-tailed distributions, such as those for transfer sizes or the ON/OFF behaviours of network users. Several models for self-similar stochastic processes exist, including Fractional Gaussian Noise and Fractional-ARIMA processes.

2.4 Web Proxy Cache Performance

Several recent research papers have explored the relationships between Web workload characteristics and Web proxy caching performance, particularly in caching hierarchies [10,12,15]. However, most of this research focuses on the frequency-domain aspect of the Web cache filter effect. For example, Doyle *et al.* [15] describe the "trickle down" effect, and conduct a detailed simulation study to quantify its impact. Fonseca *et al.* [18] study how the temporal locality structure is transformed at multiplexing and demultiplexing points in a network. Che *et al.* [12] propose a frequency-based caching hierarchy, where the lower-level cache handles high-frequency items, and the higher-level cache handles low-frequency items. Busari and Williamson [10] propose a "heterogeneous" Web proxy caching hierarchy that uses different caching policies at each level of a caching hierarchy.

Few papers explicitly address the structural changes in the request arrival process in multi-level Web caching hierarchies. Our work studies the time-domain cache filter effects using empirical and synthetic Web proxy workloads, for both a single-level Web proxy cache [4] and for multi-level Web caching hierarchies [5].

3 Experimental Methodology

In our study, we use a trace-driven simulation approach, with both empirical and synthetic Web proxy workloads. Our experimental methodology has two main steps:

- First, we start with a Web workload trace. One of our traces is an empirical workload collected from a real Web proxy server; this trace reflects user behaviour in a realistic network. We supplement this trace with several synthetically-generated workload traces. Synthetic workload generation offers greater control over workload characteristics, and allows us to study the sensitivity of our results to particular workload characteristics. It also allows us to generate traces that are as long or as short as needed for our study, without having to worry about non-stationary behaviour, which can be significant in empirical Web proxy workloads. After we generate a set

Table 1
Characteristics of Empirical Web Proxy Workload (U of S Proxy)

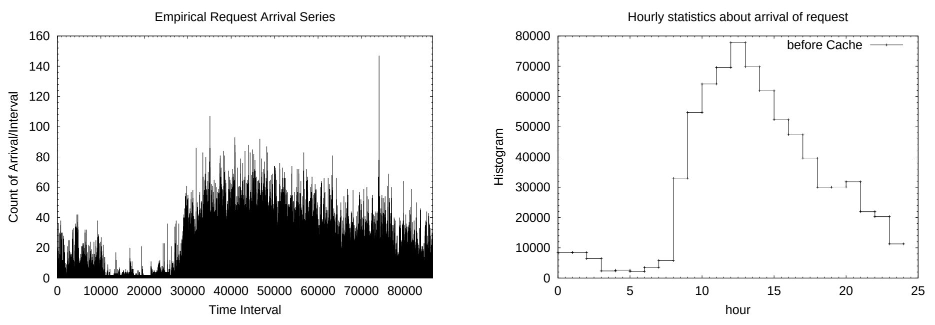
Item	Value
Trace Duration	1 day
Trace Date	Oct 17, 2001
Total Requests	755,505
Total Transferred Bytes (Mbytes)	1,087
Mean Transfer Size (bytes)	1,508
Median Transfer Size (bytes)	210
Total Documents	271,285
Unique Documents (% of requests)	35.9%
Total Bytes of Documents (Mbytes)	523
Smallest Document Size (bytes)	0
Largest Document Size (bytes)	86,399,329
Mean Document Size (bytes)	2,021
Median Document Size (bytes)	288
One-timer Documents	201,674
One-timers (% of unique documents)	74.3%
Zipf Slope	-0.8

of synthetic Web proxy workloads to use in our study, we verify that these workloads have the intended workload characteristics, and are representative of empirical Web proxy workloads.

- Second, we conduct a set of trace-driven simulation experiments, using an application-level Web proxy caching simulator and the workload traces. The output "miss" streams from the Web proxy cache simulations are used to quantify the filter effect of the cache, as a function of cache size and cache replacement policy.

3.1 Empirical Web Proxy Workload

This section describes the empirical Web proxy workload used in our study.



(a) Interval Size: 1 second (b) Interval Size: 1 hour

Fig. 2. Request Arrival Count Time Series for Empirical Web Proxy Workload

3.1.1 Overview

The empirical Web proxy workload used in our study was collected from a campus-wide Web proxy server at the University of Saskatchewan. In this paper, only a one-day access log is used as a representative example of the proxy server workload. This access log was collected on Wednesday October 17, 2001. This is the same proxy server for which long-duration (9-month) traces were analyzed in previous research [21].

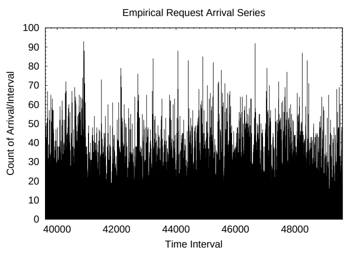
This empirical trace represents a typical one-day workload, from midnight of one day to midnight of the next. The trace contains about 750,000 requests, with request timestamps recorded at millisecond time granularity. Table 1 summarizes the characteristics of the workload.

Figure 2 shows two time series plots illustrating the characteristics of this trace. The horizontal axis shows the time of day, from midnight of one day to midnight of the next. The vertical axis shows the count process for the number of Web requests arriving in each sampling interval (1 second intervals in Figure 2(a), and 1 hour intervals in Figure 2(b)) throughout the day. Figure 2(a) shows that the arrival process is quite bursty throughout the day.

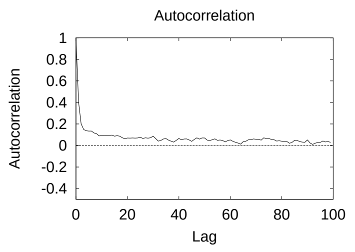
Figure 2(a) shows significant non-stationarities in the daily traffic, a behaviour that is even more evident in Figure 2(b). It is well known that Internet traffic exhibits a daily cyclic pattern, based on the "working hours" for human users. Since this workload is from a university proxy cache, most of the daily Web traffic (71%) occurs between 9am and 6pm.

3.1.2 Self-Similar Arrival Process

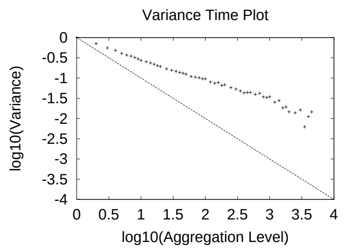
The next analysis focuses on the three-hour "busy period" of the empirical workload trace from 11am to 2pm. This period contains 217,159 requests, representing about 30% of the daily Web request traffic. For this period, we



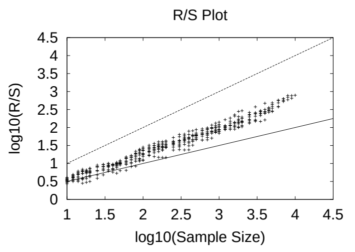
(a) Time Series Plot



(b) Autocorrelation Function



(c) Variance-Time Plot



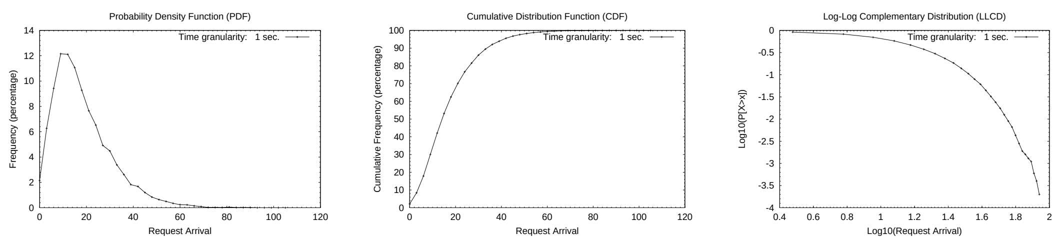
(d) R/S Pox Plot

Fig. 3. Evidence of Self-Similarity for Empirical Web Proxy Workload

hypothesize that the arrival count process (for 1 second intervals) is stationary (see Figure 3(a)). We use this portion of the trace to characterize the request arrival process, and to test for long-range dependence and self-similarity [20] in the arrival count process. We use the standard statistical analysis techniques described by Leland et al. [20], namely the autocorrelation function, the variance-time plot, and the rescaled adjusted range statistic (R/S).

Figure 3 presents the results from the tests for network traffic self-similarity. Figure 3(a) shows the time series under consideration. Figure 3(b) shows the autocorrelation function for this time series. The hyperbolic decay is indicative of self-similarity, though the length of the time series (10,800 data points) is somewhat short to be sure. Figure 3(c) shows a variance-time plot for this time series. The points plotted in the graph show a straight line behaviour with a slope significantly flatter than -1 (the solid line in the graph). This graph suggests a slowly-decaying variance for the aggregated time series, another indication of self-similarity. Finally, Figure 3(d) shows an R/S pox plot for this data set. The slope of this scatter plot can be used to estimate the Hurst parameter H characterizing the degree of self-similarity in this data set. The R/S plot provides a Hurst parameter estimate of $H = 0.74$, again suggesting that the arrival count process is self-similar.

Further detail on the traffic arrival process is provided in Figure 4. Figure 4(a) shows the marginal distribution (i.e., frequency histogram, or probability density function, PDF) of the traffic arrival process from Figure 3(a). The average arrival rate is 20 requests per second, but there is a significant skew to the distribution. Figure 4(b) shows the cumulative distribution function for the



(a) PDF (b) CDF (c) LLCD

Fig. 4. Request Arrival Count Process for Empirical Web Proxy Workload

arrival process, while Figure 4(c) shows a log-log complementary distribution (LLCD) plot that illustrates the tail behaviour of the distribution. The downward curvature of Figure 4(c) suggests that the marginal distribution for the count arrival process is *not* heavy-tailed, though the arrival process does appear to be self-similar.

This workload serves as the input to a trace-driven simulation study of cache filter effects in Section 4.

3.2 Synthetic Web Proxy Workloads

For synthetic Web workload generation, there are two workload parameters of interest: Zipf slope, and request arrival process. The Zipf slope refers to the slope of the Zipf-like document popularity profile in the input Web workload. This slope affects the magnitude of the Web cache filtering effect, because it determines the relative skew to the distribution of references amongst the popular documents. In particular, a steep Zipf slope tends to produce a high cache hit ratio, while a flat Zipf slope does not. The request arrival process refers to the timestamps generated for the Web client requests. We consider a self-similar arrival process that provides a realistic representation of the Web request arrival process. These workload factors are summarized in Table 2.

Table 2
Experimental Factors and Levels for Studying Cache Filter Effects

Type	Factor	Levels
Workload	Zipf Slope	0.5, 0.6, 0.7, 0.8, 0.9, 1.0
Workload	Arrival Process	Self-Similar
Experimental	Cache Size (MB)	1, 4, 16 . . . 1024
Experimental	Cache Replacement Policy	RAND, FIFO, LRU, LFU, GDS

In our work, a Web proxy workload modeling tool called *ProWGen* (Proxy Workload Generation) [11] is used to synthesize Web proxy workload traces.

ProWGen captures the salient characteristics of Web proxy workloads: one-time referencing, Zipf-like document popularity, heavy-tailed file size distribution, and temporal locality. These characteristics are deemed relevant to Web proxy cache performance [9,11], and are easy to generate and analyze using the *WebTraff* tool [22]. By design, the synthetic workloads differ in the slope for the Zipf-like document popularity distribution, which has a significant influence on the cache hit ratio [8,9,27].

For each of the synthetic traces generated, multiple arrival-time time series are used. Note that the generation of the traffic arrival process (i.e., the timestamps on the Web document requests) is independent of the techniques used to generate the Web document requests (i.e., ProWGen's file popularity and temporal locality models).

The resulting synthetic workloads are used in the latter half of the paper to investigate cache filter effects in a two-level Web proxy caching hierarchy.

3.3 Web Proxy Cache Simulation

In the Web cache simulation step, two experimental factors are used: cache size, and cache replacement policy. The cache size determines the maximum number of Web content bytes that can be held in the cache at one time. The cache replacement policy determines what objects to remove from the cache when more space is needed to store an incoming object. Five cache replacement policies are considered: removing objects at random (RAND), removing objects in the order in which they arrived (First-In-First-Out, FIFO), removing objects based on recency of use (Least-Recently-Used, LRU), removing unpopular objects (Least-Frequently-Used, LFU), and removing large objects (Greedy-Dual-Size, GDS). These experimental factors are also summarized in Table 2.

In a set of simulation experiments in this work, the empirical and synthetic Web workloads (timestamped series of Web document requests) are provided as input to the Web proxy cache simulator. The network topology modeled is similar to that shown in Figure 1. The simulator allows configuration of the Web proxy cache size and the cache replacement policy (i.e., which document(s) to remove when the cache is full). The simulator generates as output the cache hit ratios for the experiment, and a timestamped series indicating the requests that result in cache misses. The latter output is called the *filtered request stream*, and is used in our subsequent traffic analyses.

The experiments use cache hit ratio and byte hit ratio as the primary performance metrics for cache performance, and the mean and variance of the filtered request arrival process to characterize cache filter effects.

4 Simulation Results: Empirical Workload

4.1 Overview of Cache Filtering Effects

The first simulation experiment illustrates the general impacts of the proxy cache on the Web workload. These cache filter effects are shown in Figure 5.

Figure 5(a) shows the request arrival count time series for the filtered and unfiltered request streams, as a function of time of day. For clarity of presentation, this plot shows the entire workload trace, with request counts sampled over 30 minute intervals. The graph clearly shows that the presence of the Web cache reduces both the peak and the mean rate of the request arrival process. The larger the cache size, the more pronounced the filter effect. These results are as expected.

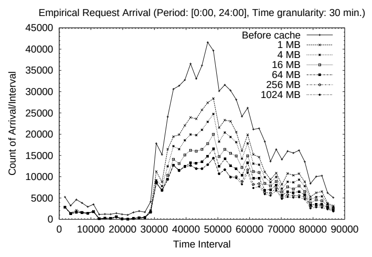
Figure 5(b) shows the same type of plot, but just for the three-hour busy period of the trace (11am to 2pm). In this plot, the request counts are computed over 5 minute intervals. Again, the plot shows a consistent reduction in the mean and peak rate of requests, as the cache size is increased.

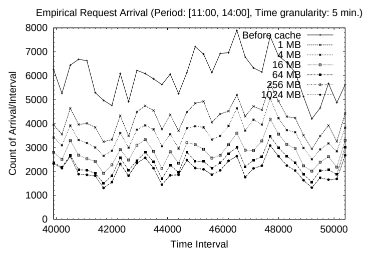
Figure 5(c) shows the average document hit ratio in the cache, as a function of time of day, for different cache sizes. Other than the erratic behaviour in the early morning hours (say, 2am to 7am) when few clients are using the cache, the cache hit ratio is relatively stable throughout the day, reflecting "steady state" cache performance for the workload considered. Note that these results are plotted using the average cache hit ratio over each 30 minute interval of the trace. The cache is initially empty at midnight, and proceeds to fill throughout the day, invoking the cache replacement policy as needed to manage the contents of the cache. Overall, the cache hit ratio tends to increase as the cache size is increased (as expected).

Figure 5(d) shows the corresponding cache hit ratio results for the busy portion of the day from 11am to 2pm. In this plot, the cache hit ratios are computed over 5 minute intervals, with the cache initially empty at midnight. The cache hit ratio clearly increases with the cache size, and is relatively stable throughout the busy period. These observations suggest that the filtered request arrival process for the busy period is likely a stationary process.

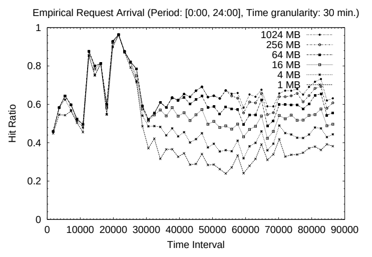
4.2 Self-Similarity

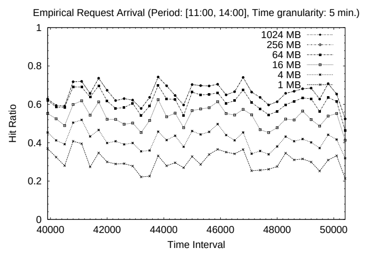
The purpose of the next analysis is to see if the filtered request stream after the Web proxy cache still has the same self-similar properties of the input request





(a) Full Trace Time Series (b) Busy Period Time Series





(c) Full Trace Hit Ratio (d) Busy Period Hit Ratio

Fig. 5. Illustration of Cache Filter Effects on the Empirical Web Proxy Workload

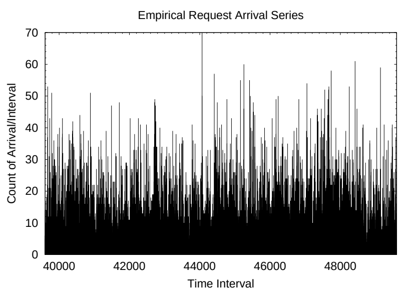
stream. As an example, we consider the simple case with a Least-Frequently-Used (LFU) cache replacement policy, with a cache size of 16 MB. We use the same statistical analysis techniques in Figure 3. The results in Figure 6 show that the self-similar characteristics remain present in the filtered request stream. The Hurst parameter is estimated as $H = 0.71$. Our investigations suggest that the self-similar property of the arrival count process is not altered by the presence of the Web proxy cache.

4.3 Effect of Cache Size

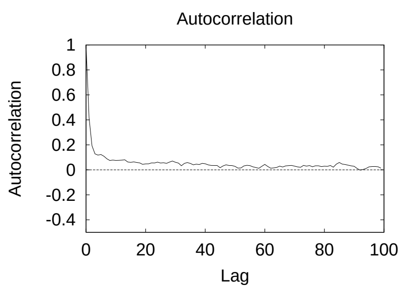
The next experiment studies the arrival count process for the filtered request stream, as a function of cache size. We focus on the 11am-2pm busy period of the trace, using arrival counts per one second interval.

Table 3 summarizes these simulation results. Clearly, the presence of the cache reduces both the mean and the standard deviation of the arrival count process after the cache, though the impact on the mean is more pronounced. The larger the cache, the greater this filtering effect. The corresponding cache hit ratios for different cache sizes are also shown in Table 3.

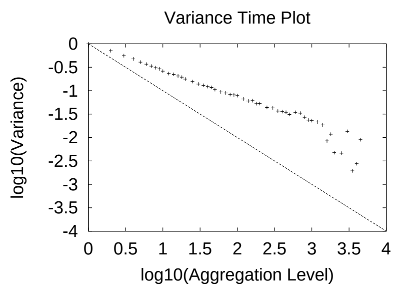
The characteristics of the filtered arrival process are shown in Figure 7. Figure 7(a) shows the marginal distribution (i.e., PDF) of the filtered request streams, for different cache sizes. For ease of reference, the unfiltered request stream is shown using a cache size of 0 MB. The corresponding cumulative



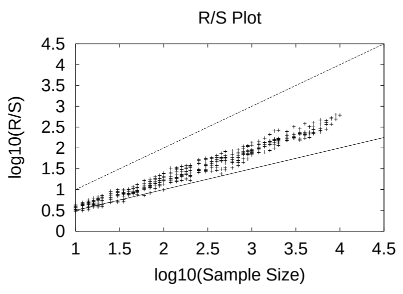
(a) Time Series Plot



(b) Autocorrelation Function



(c) Variance-Time Plot



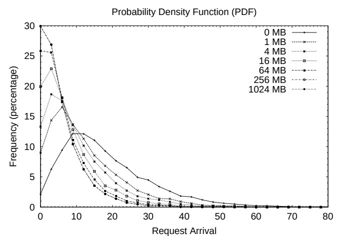
(d) R/S Pox Plot

Fig. 6. Evidence of Self-Similarity for Filtered Web Proxy Workload

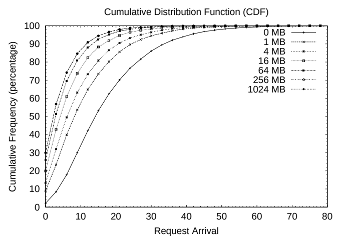
distribution functions (CDF) for the arrival count processes are shown in Figure 7(b), and the LLCD plots in Figure 7(c). In general, the filter effect of the cache increases with cache size.

Table 3
Simulation Results for Different Cache Sizes (Empirical Workload, LFU Policy)

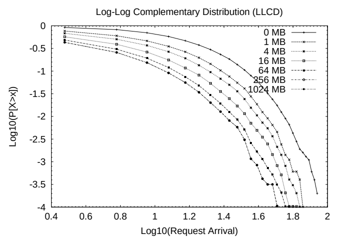
Arrival Count Statistics	Before Cache	Cache Size (MB)					
		1	4	16	64	256	1024
Mean	20.27	14.02	11.78	9.38	7.75	6.88	6.88
StdDev	12.41	10.24	9.22	7.85	6.72	6.09	6.09
Hit Ratio	-	33.5%	42.9%	52.5%	59.1%	63.0%	64.1%



(a) PDF



(b) CDF



(c) LLCD

Fig. 7. Effect of Cache Size on Filtered Arrival Count Process (Empirical Workload, LFU Policy)

In Figure 7(a), the filter effect of the cache manifests itself in several ways. First, the main “hump” of the marginal distribution moves to the left, reflecting the decrease in the mean arrival rate. Second, the height of the distribution at or near the origin tends to increase, since a large cache produces many one-second intervals with few (or even zero) arriving requests. Third, the distribution tends to decay more quickly (i.e., it has a shorter tail), reflecting the lower variance in the resulting arrival process. The latter two effects together tend to produce a taller and narrower marginal distribution, again reflecting lower variance in the filtered arrival process.

4.4 Effect of Cache Replacement Policy

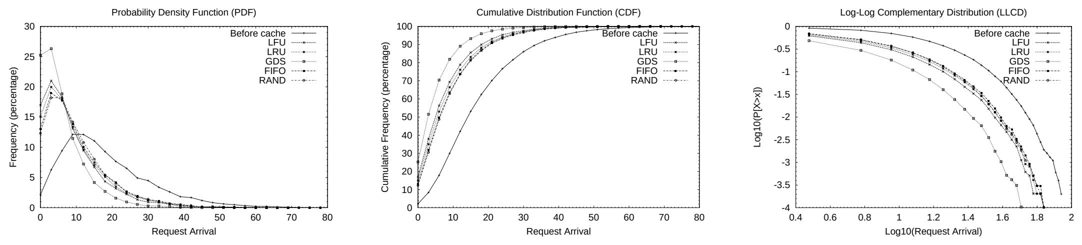
The next experiment looks at the sensitivity of the cache filter effect to the cache replacement policy used. The cache replacement policy determines which document(s) to remove from the cache when more space is needed to store an incoming document. Different cache replacement criteria have been proposed in the literature, such as recency-based, frequency-based, size-based, and so on. We consider five example policies in our study, namely RAND, FIFO, LRU, LFU, and GDS. Further details on these policies can be found in the literature [9,10,29].

Table 4 summarizes the statistical characteristics of the filtered request arrival process for an 8 MB cache. As expected, the document hit ratio for the GDS policy (60.6%) is higher than for the other policies. The GDS policy tries to keep small documents (and therefore more of them) in the cache, by associating a weight $H = \frac{1}{s}$ with each document, where s is the size of the document in bytes. FIFO and RAND have less of a filtering effect on the workload, since their “zero knowledge” approach produces a lower cache hit ratio.

Figure 8 shows the impact of the selected cache replacement policies on the workload characteristics. Examining the plots shows that the filter effects for the LFU and LRU policies are similar. The GDS policy has the most pronounced impact on the request arrival count process, because of its higher cache hit ratio.

4.5 Summary of Results

This section focused on the time-domain analysis of cache filter effects for an empirical Web workload trace. In general, increasing the cache size significantly reduces the peak and the mean of the arrival rate for the filtered request stream, but the impact on the variance is less pronounced. The variance-to-mean ratio of the filtered request stream tends to increase. The next section



(a) PDF (b) CDF (c) LLCD

Fig. 8. Effect of Cache Replacement Policy on Filtered Arrival Count Process (Empirical Workload, 8 MB Cache)

Table 4
Simulation Results for Different Cache Replacement Policies (Empirical Workload, 8 MB Cache)

Arrival Count	Before	Cache Replacement Policy				
Statistics	Cache	RAND	FIFO	LRU	LFU	GDS
Mean	20.27	11.75	11.70	11.05	10.34	7.63
StdDev	12.41	8.82	9.13	8.85	8.39	6.33
Hit Ratio	-	43.0%	43.8%	46.4%	48.6%	60.6%

generalizes these observations to a broader set of synthetically-generated Web workload traces.

5 Simulation Results: Synthetic Workloads

This section presents additional simulation results for synthetically generated Web proxy cache workloads. Many of the foregoing observations about cache filter effects for the empirical workload apply equally well for synthetic workloads. These observations (e.g., reduction in mean and variance for the filtered request stream; preservation of self-similarity; leftward movement of the arrival count distribution with larger cache sizes; and minimal dependence on cache replacement policy) are not repeated here, since they are documented in prior work [5]. Instead, we focus on the impact of Web workload characteristics, such as the Zipf slope and the request arrival process.

5.1 Effect of Web Workload Characteristics

Our simulation experiments with synthetic workloads study the impact of specific Web workload characteristics on cache filter effects. Three different

Web proxy workload traces are generated for this particular set of experiments.

Table 5 summarizes the statistical characteristics of the synthetic traces used. Each trace has approximately 220,000 requests, with an average arrival rate of about 20 requests per second, similar to the 3-hour busy period of the empirical workload in Section 4.

By design, the three synthetic workloads differ in the slope for the Zipf-like document popularity distribution, which has a significant influence on the cache hit ratio [8,9,27]. Trace B has a Zipf slope of 0.8, which closely matches that of the empirical trace used. Trace A has a flatter Zipf slope (0.6), which implies a lower cache hit ratio is expected for this trace. Trace C has a steeper Zipf slope, meaning that this trace will produce higher cache hit ratios, and thus a more pronounced cache filtering effect.

For each of these three traces, three different traffic arrival processes were generated: a short-range dependent arrival process with $H = 0.5$, a self-similar process with $H = 0.75$ (similar to the empirical workload), and a self-similar process with Hurst parameter $H = 0.9$. Recall that the generation of the traffic arrival process is independent of the techniques used to generate the Web document requests. Thus for a given Hurst value H, the distribution of the arrival count process remains the same even when the Web document popularity distribution is changed. For a given document popularity distribution, the arrival count process varies for $H = 0.5$, $H = 0.75$, and $H = 0.9$. The $H = 0.9$ workload has the highest degree of self-similarity, and thus the longest tail to the arrival count distribution.

The three synthetic traces combined with the three traffic arrival models produce nine synthetic traces with a wide range of workload characteristics. These nine traces are used in the simulations to assess the robustness of our observations about the filtered Web proxy workload characteristics.

Table 6 and Table 7 summarize the simulation results for one representative example of the synthetic workloads ($H = 0.9$, Zipf slope 0.8). Table 6 shows the characteristics of the filtered request stream, as a function of cache size, for an LFU replacement policy. Table 7 shows the characteristics of the filtered request stream, as a function of cache replacement policy, for an 8 MB cache. The characteristics of the filtered request stream are similar to those for the empirical trace in Section 4.

Figure 9 shows the graphical characteristics of the filtered request arrival process. For space reasons, only the results for $H = 0.75$ are shown. The results are consistent for the other Hurst parameter values considered in our experiments.

The qualitative behaviour in Figure 9 follows that observed for the empirical

Table 5
Characteristics of Synthetic Web Proxy Workloads

Item	Trace A	Trace B	Trace C
Trace Duration	3 hours	3 hours	3 hours
Total Requests	225,042	218,845	225,697
Total Transferred Bytes (Mbytes)	2,144	1,871	1,654
Mean Transfer Size (bytes)	9,991	8,967	7,683
Median Transfer Size (bytes)	3,552	3,474	3,285
Total Documents	70,254	70,870	70,951
Unique Documents (% of requests)	31.2%	32.4%	31.4%
Total Bytes of Documents (Mbytes)	793	798	799
Smallest Document Size (bytes)	34	34	34
Largest Document Size (bytes)	12,382,599	12,382,599	12,382,599
Mean Document Size (bytes)	11,830	11,808	11,806
Median Document Size (bytes)	3,817	3,816	3,815
One-timer Documents	48,942	49,558	49,638
One-timers (% of documents)	69.7%	69.9%	70.0%
Zipf Slope	**-0.6**	**-0.8**	**-1.0**

Table 6
Simulation Results for Different Cache Sizes (Synthetic Workload, $H = 0.9$, $Z = 0.8$, LFU Policy)

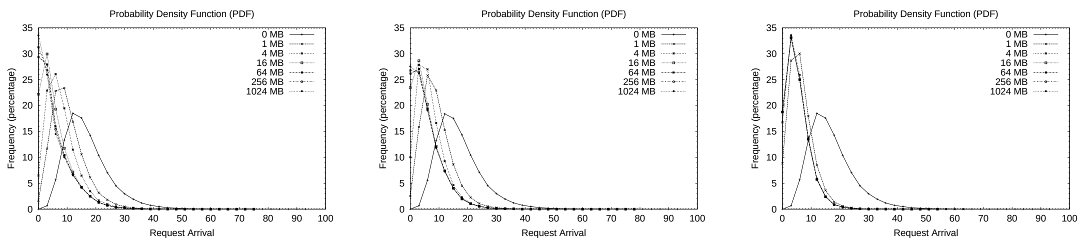
Arrival Count	Before	Cache Size (MB)					
Statistics	Cache	1	4	16	64	256	1024
Mean	23.60	12.67	9.75	7.83	7.65	7.64	7.64
StdDev	16.80	9.01	8.14	8.28	8.32	8.32	8.32
Hit Ratio	-	46.3%	58.7%	66.8%	67.6%	67.6%	67.6%

Web proxy trace. Larger cache sizes produce a leftward shift of the arrival count distribution for the filtered request stream, producing an increase in its peak value at the origin.

The important observation is that the leftward shift of the distribution with increasing cache size is very *gradual* in Figure 9(a) for $Z = 0.6$, and very *sudden* in Figure 9(c) for $Z = 1.0$. In other words, the filter effect of the cache is (as expected) more pronounced (particularly at smaller cache sizes) when

Table 7
Simulation Results for Different Cache Replacement Policies (Synthetic Workload, $H = 0.9$, $Z = 0.8$, 8 MB Cache)

Arrival Count	Before	Cache Replacement Policy				
Statistics	Cache	RAND	FIFO	LRU	LFU	GDS
Mean	23.60	9.62	9.35	8.84	8.48	8.03
StdDev	16.80	8.36	8.24	8.06	8.12	8.23
Hit Ratio	-	59.2%	60.4%	62.5%	64.1%	66.0%



(a) $Z = 0.6$ (b) $Z = 0.8$ (c) $Z = 1.0$

Fig. 9. Effect of Cache Size on Filtered Arrival Count Process ($H = 0.75$, LFU Policy)

the Zipf slope is steep.

5.2 Summary of Results

This section studied cache filter effects for synthetic traces. The results demonstrate the relationships between input workload characteristics and the characteristics of the output filtered request stream for different Zipf slopes and different request arrival processes.

6 Modeling Web Cache Filter Effects

This section discusses a parameterizable mathematical model for characterizing the request arrival count distribution both before and after a Web proxy cache. Prior experience with network traffic modeling suggests that a Gamma distribution may be suitable for modeling the (filtered or unfiltered) arrival count distribution for Web workloads, since the shape of the distributions in Figure 7 and elsewhere are reminiscent of the Gamma distribution. Section 6.1 provides some background on the Gamma distribution, while Section 6.2 validates the Gamma distribution model on the empirical workload. Section 6.3

extends these results to a broad set of synthetically generated workloads.

6.1 Background

The general formula for the Gamma distribution probability density function (PDF) is:

$$f(x) = \frac{(\frac{x-\mu}{\beta})^{\gamma-1} e^{(-\frac{x-\mu}{\beta})}}{\beta\,\Gamma(\gamma)} \qquad x \geq \mu; \quad \gamma, \beta > 0 \tag{1}$$

where γ is the *shape* parameter, μ is the *location* parameter, β is the *scale* parameter, and Γ is the Gamma function

$$\Gamma(a) = \int_{0}^{\infty} t^{a-1} e^{-t} dt \tag{2}$$

The case where $\mu = 0$ and $\beta = 1$ is called the standard Gamma distribution. The equation for the standard Gamma distribution reduces to:

$$f(x) = \frac{x^{\gamma-1} e^{-x}}{\Gamma(\gamma)} \qquad x \geq 0; \quad \gamma > 0 \tag{3}$$

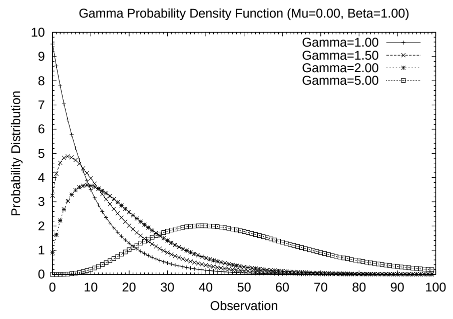
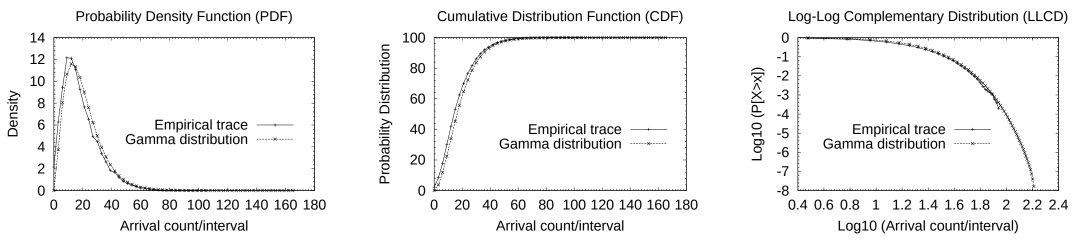


Fig. 10. Gamma Probability Density Function

Figure 10 shows examples of the probability density function for the standard Gamma distribution with different choices of shape parameter γ. As γ decreases, the center of gravity of the distribution moves to the left, the peak value of the curve increases, and the tail of the curve decreases more quickly. If $\gamma \leq 1$, then the distribution is monotonically decreasing. These behaviours are similar to those for the empirical and synthetic request arrival count processes in our study, suggesting the suitability of the Gamma distribution for our traffic modeling purposes.



(a) PDF (b) CDF (c) LLCD

Fig. 11. Gamma Distribution Model for Input Request Arrival Count Distribution (Empirical Workload, $\gamma = 2.67$, $\beta = 7.60$)

6.2 Modeling the Empirical Request Arrival Count Distribution

Our modeling efforts start with the empirical Web proxy workload. In our initial experiments, we assume that the *location* parameter for the Gamma distribution is $\mu = 0$. Let M and D denote the mean and standard deviation, respectively, of the request arrival count process in the Web workload.

The shape parameter γ and the scale parameter β of the Gamma distribution can be determined using *Maximum Likelihood Estimates* (MLE), namely:

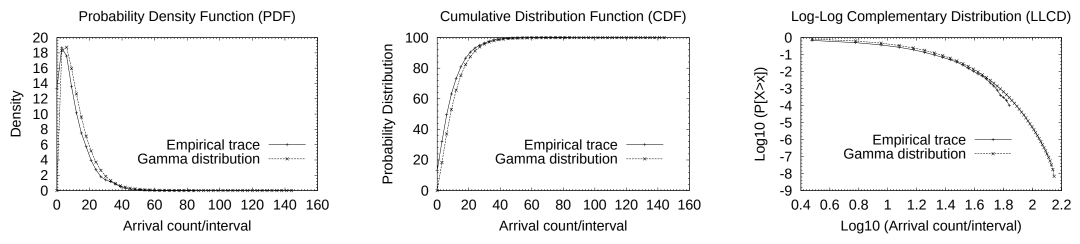
$$\hat{\gamma} = (\frac{M}{D})^2 \tag{4}$$

and

$$\hat{\beta} = \frac{D^2}{M} \tag{5}$$

Figure 11 illustrates the characteristics of the request arrival count distribution for the empirical workload, along with a Gamma model fit to the distribution. The parameters γ and β of the Gamma distribution ($\gamma = 2.67$, $\beta = 7.60$) were determined using MLE. Figure 11 shows that the Gamma distribution provides a good visual fit of the distribution, for both the body and the tail of the distribution.

Figure 12 shows similar plots for the filtered request arrival process after the cache. This particular plot illustrates the results for a 4 MB LFU cache at the Web proxy. The Gamma distribution with $\gamma = 1.63$ and $\beta = 7.22$ provides a good visual fit to the traffic arrival distribution, in both the body and the tail. Again, the parameters of the Gamma distribution were determined using MLE.



(a) PDF (b) CDF (c) LLCD

Fig. 12. Gamma Distribution Model for Filtered Request Arrival Count Distribution (Empirical Workload, $\gamma = 1.63$, $\beta = 7.22$)

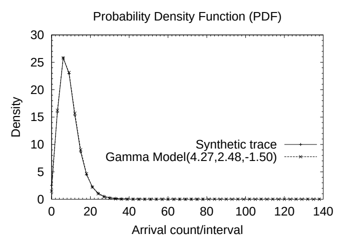
6.3 Modeling the Synthetic Request Arrival Count Distribution

To generalize our Gamma distribution modeling results, we have conducted many additional simulation experiments with synthetic workloads. These experiments have varied the trace duration (from 250,000 to 1 million requests), the request arrival rate (from 20 to 60 requests per second), the request arrival process (from $H = 0.5$ to $H = 0.9$), the Zipf slope (from 0.5 to 1.0), and the cache size (from 1 MB to 1 GB). Across the broad range of workloads studied, we have found that Gamma location parameter $\mu = -1.5$ provides the basis for consistently good modeling results.

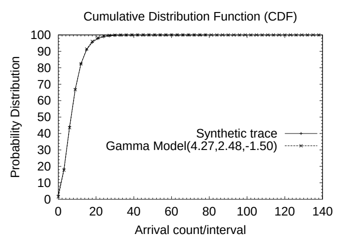
In all simulation scenarios studied, the Gamma distribution (with γ and β parameters determined using the MLE technique) provides a good fit to the request arrival count distribution for both the input and the filtered request streams. We thus believe that the Gamma distribution provides a robust and flexible means of characterizing the request count process for workloads in Web caching performance studies.

One particular set of simulation experiments explored the relationship between Zipf slope, cache size, and the γ and β parameters in the Gamma distribution. We used least-squares linear regression to determine the relationship between the Zipf slope and the mean arrival rate for the filtered request stream, for different cache sizes. The impact of the Zipf slope on the mean arrival rate is pronounced for small cache sizes, but negligible when the cache size is large. A similar qualitative relationship is observed between the Zipf slope and the standard deviation of the filtered request arrival count process, though the regression coefficients differ.

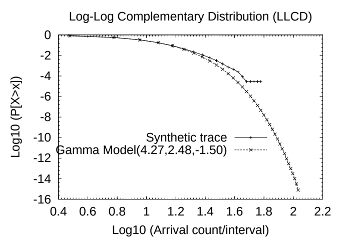
Understanding these relationships helps build intuition as to how γ and β change with the cache size. In general, as the cache size is increased, the γ parameter decreases and the β parameter increases (for a given Zipf slope, and a fixed value of the location parameter μ). This behaviour is consistent with the leftward movement of the body of the distribution as the cache hit ratio



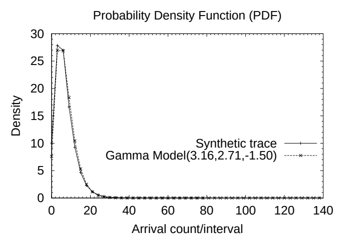
(a) PDF (1 MB)



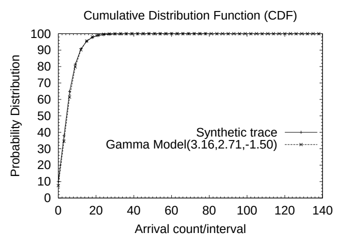
(b) CDF (1 MB)



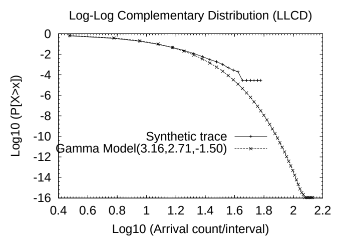
(c) LLCD (1 MB)



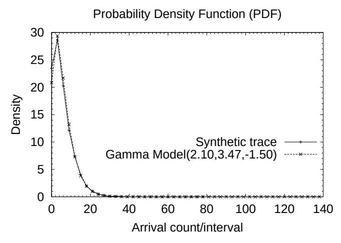
(d) PDF (4 MB)



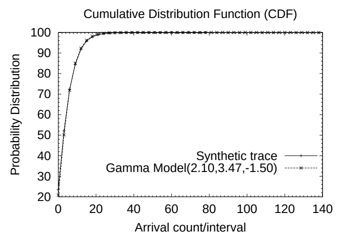
(e) CDF (4 MB)



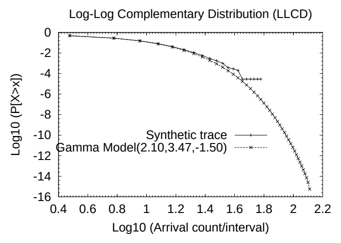
(f) LLCD (4 MB)



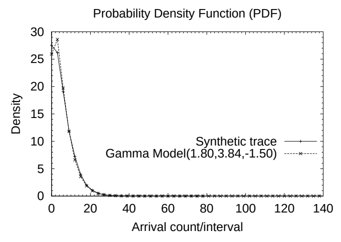
(g) PDF (16 MB)



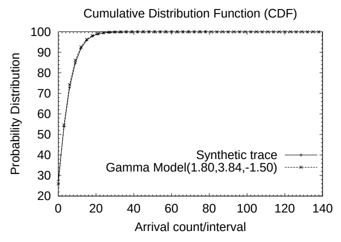
(h) CDF (16 MB)



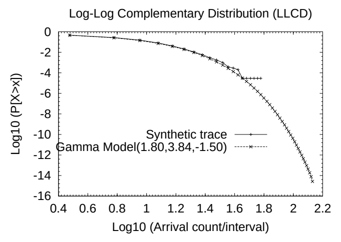
(i) LLCD (16 MB)



(j) PDF (1024 MB)



(k) CDF (1024 MB)



(l) LLCD (1024 MB)

Fig. 13. Gamma Distribution Models for Filtered Request Arrival Count Distribution (Synthetic Workload: 20 req/sec, $H = 0.75$, $Z = 0.80$, LFU Cache, Gamma Model (γ, β, μ))

increases. In the scenarios studied, the changes in γ and β are monotonic with cache size, asymptotically approaching a limit when the cache size is large.

Examples of these results are illustrated in Figure 13. These results are for a self-similar request arrival process (20 requests per second, $H = 0.75$) for a workload with a Zipf slope of 0.8, and the LFU cache replacement policy. The rows of graphs in Figure 13 show the Gamma distribution modeling results for cache sizes ranging from 1 MB (top row) to 1024 MB (bottom row). In all cases, the MLE estimates provide good Gamma distribution fits to the simulation results. These results provide the basis for Web workload modeling throughout a Web caching hierarchy, which we undertake in the next section.

7 Simulation Results: Aggregate Web Workload

This section addresses the larger challenge of characterizing workloads throughout a Web proxy caching hierarchy. As mentioned in Section 2.1, multi-level Web proxy caching systems have been used to improve the performance and scalability of the Internet. This simulation experiment quantifies the filter effects of a Web cache on the request arrival process, for synthetically-generated aggregate Web client workloads. For simplicity, we consider only a two-level Web proxy caching hierarchy, as shown in Figure 14. Assuming that λ_1 and λ_2 represent the Web request arrival processes (from clients) entering the child-level caches, we are interested in the filtered arrival processes λ_1' and λ_2' after these caches. We are also interested in how they multiplex to form λ_3, which itself is transformed into λ_3' before entering the Internet.

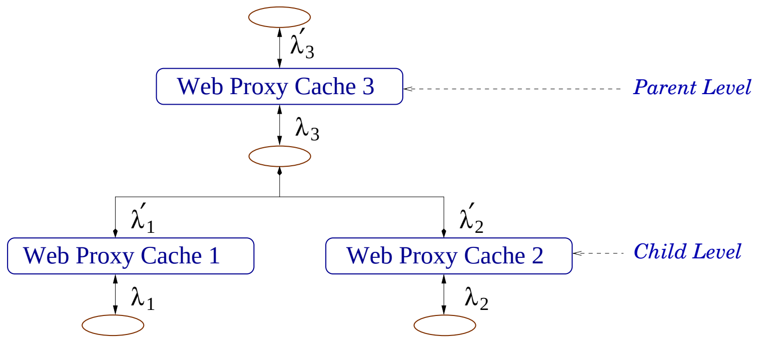


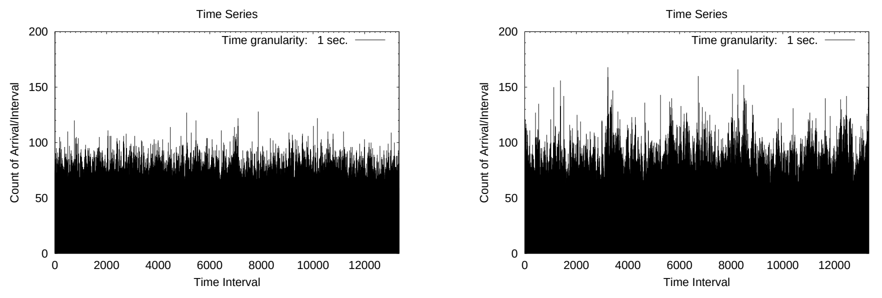
Fig. 14. Example of a Two-level Web Proxy Caching Hierarchy

We start by characterizing the input workloads λ_1 and λ_2 offered to the child-level caches, and proceed to analyze the filter effects as the workload progresses to the Internet as λ_3'. Section 7.1 describes the input workloads and the filter effects of the first-level caches. Section 7.2 studies the aggregation of the filtered request streams to form λ_3. Finally, Section 7.3 shows how to model the aggregate stream λ_3 with a Gamma distribution, using knowledge of the input workloads and cache filter effects.

7.1 Workload Modeling and Analysis

We use synthetically-generated Web proxy workloads to represent the input workloads λ_1 and λ_2 for our simulation. To demonstrate the generality of our analysis, we consider "heterogeneous" input workloads, in the sense that the workloads λ_1 and λ_2 differ in Zipf slope, Hurst parameter, and mean arrival rate.

The characteristics of the synthetic Web proxy workloads with self-similar arrival processes are illustrated in Figure 15.



(a) Trace 1: $H = 0.70$, $Z = 0.75$ (b) Trace 2: $H = 0.80$, $Z = 0.80$

Fig. 15. Synthetic Self-Similar Workload Traces Used in Simulations

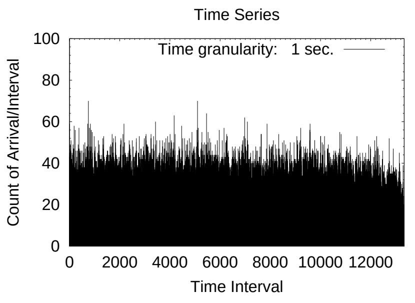
These workload traces are provided as input to the Web cache simulator at the first level of the Web caching hierarchy. In this case study, we use the GDS replacement policy, with an 8 MB cache size for both caches at the first level of the caching hierarchy. The filter effects of the child caches produce workloads λ'_1 and λ'_2, which we call the filtered workloads.

The time series of request arrival processes for the filtered request streams λ'_1 and λ'_2 are depicted in Figure 16. Other than the end effects evident in these plots, the filtered request streams appear to be stationary. We restrict our attention to the stationary portion of these traces.

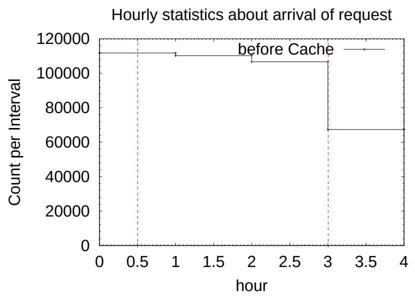
Figure 17 shows the characteristics of the filtered workload trace λ'_1. These results show the initial impact of the Web cache.

Figure 18 is used to test for self-similarity in the filtered request arrival process. The hyperbolic decay of the autocorrelation function in Figure 18(a) indicates self-similarity. The variance-time plot in Figure 18(b) shows a slowly-decaying variance for the filtered time series. Finally, the slope of the R/S plot in Figure 18(c) provides a Hurst parameter estimate of $H \approx 0.699$ (very close to 0.70, the degree of self-similarity of the input workload).

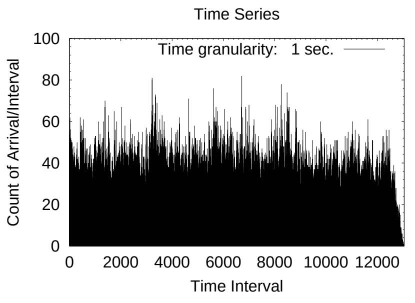
All these observations suggest that the arrival process of the filtered workload λ'_1 is self-similar. The same observations apply for filtered request stream λ'_2. Its analysis is omitted for space reasons.



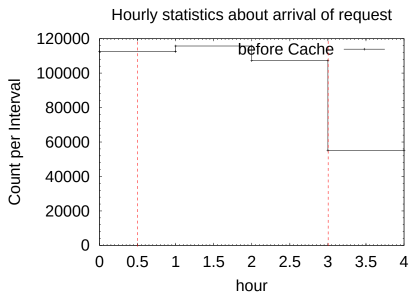
(a) λ'_1: Interval Size=1 second



(b) λ'_1: Interval Size=1 hour

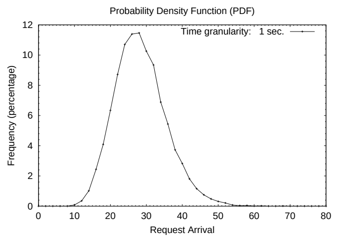


(c) λ'_2: Interval Size=1 second

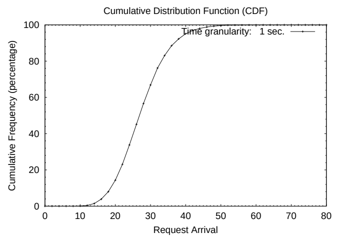


(d) λ'_2: Interval Size=1 hour

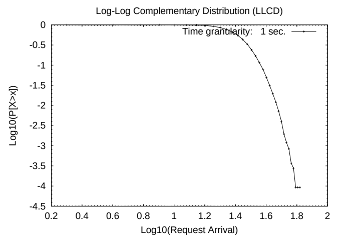
Fig. 16. Time Series Plots of Request Arrival Count Processes for Filtered Workloads λ'_1 and λ'_2



(a) PDF

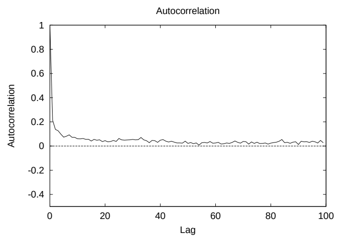


(b) CDF

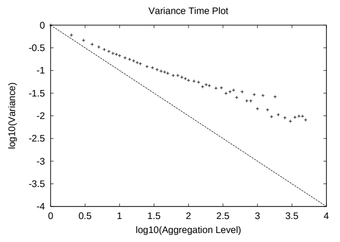


(c) LLCD

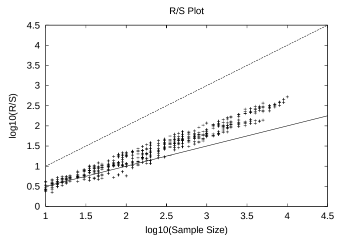
Fig. 17. Characteristics of Filtered Workload λ'_1



(a) Autocorrelation

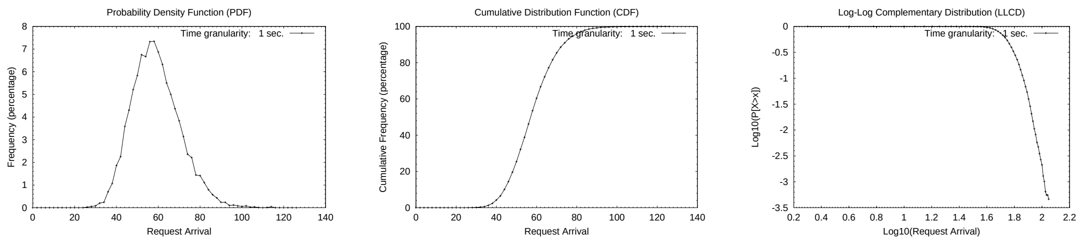


(b) Variance-Time Plot



(c) R/S Pox Plot

Fig. 18. Evidence of Self-Similarity in Filtered Workload λ'_1



(a) PDF (b) CDF (c) LLCD

Fig. 19. Characteristics of Aggregate Request Arrival Process λ_3

7.2 Superposition of Web Workloads

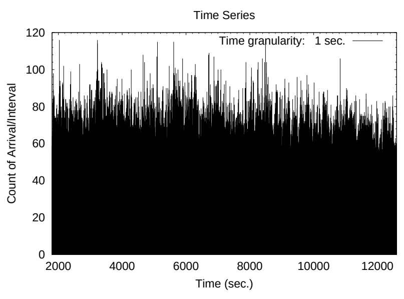
Our investigations so far demonstrate that the first-level cache changes the structural characteristics of the workload. However, it does not remove the self-similar property of the workloads. This section focuses on the superposition of Web workload streams (with self-similar arrival) in time-domain. Understanding the statistical multiplexing behaviour is important since many networks rely on this principle.

In this part of the study, the workload traces λ_1' and λ_2' (that is, misses from the lower level proxies) are statistically multiplexed in the time-domain, resulting in an aggregate workload λ_3. Here we consider portions of each trace that cover the identical time period in order to get a stationary aggregate workload trace. We assume that λ_1' and λ_2' are independent.

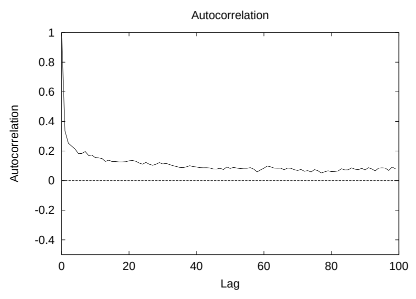
The characteristics of the aggregate workload are illustrated in Figure 19. Figure 19(a) shows the marginal distribution (i.e., frequency histogram, or probability density function, PDF) of the traffic arrival process from Figure 20(a). As expected, the mean arrival rate for the aggregate arrival process is equal to the sum of those from the two individual input processes. Figure 19(b) shows the cumulative distribution function for the arrival process, while Figure 19(c) shows a log-log complementary distribution (LLCD) plot that illustrates the tail behaviour of the distribution.

Figure 20 presents the results from the examination of self-similarity for the aggregate workload. Again, we use the autocorrelation function, the variance-time plot, and the rescaled adjusted range statistic (R/S). Figure 20(a), (b) and (c) demonstrate all of the important indicatives of self-similarity, i.e. the hyperbolic decay of autocorrelation function; a slowly-decaying variance-time plot, etc.

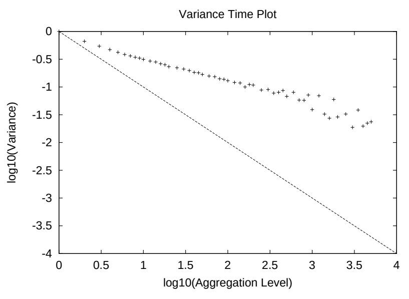
The R/S plot provides a Hurst parameter estimate of $H \approx 0.76$, suggesting that the aggregate arrival count process is self-similar. These observations are in close agreement with theoretical results [26], which indicate that the



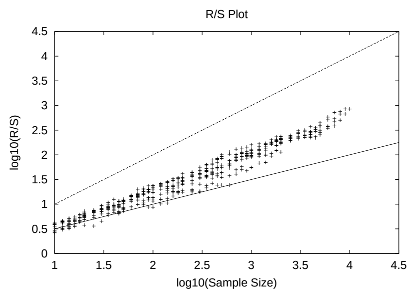
(a) Time Series



(b) Autocorrelation Function



(c) Variance-Time Plot



(d) R/S Pox Plot

Fig. 20. Evidence of Self-Similar Aggregate Request Arrival Process λ_3: $H \approx 0.76$

Hurst parameter of the aggregate stream should be the maximum of the Hurst parameters of the input streams. In addition, the variance-to-mean ratio of the aggregate process should be the weighted average of those from the input streams [26].

These analyses demonstrate that the superposition of Web traffic streams does not smooth the traffic. Multiplexing bursty data streams tends to produce a bursty aggregate stream. The aggregate Web workload is then forwarded to the next level of the Web proxy caching hierarchy.

7.3 Modeling of Aggregate Workload

Finally, we address the issue of modeling aggregate Web workload in a Web caching hierarchy. Following the approach in Section 6, we propose two Gamma probability density functions $f_1'(x)$ and $f_2'(x)$ for the Web request arrival count λ_1' and λ_2', namely

$$f_1'(x) = \frac{(\frac{x-\mu_1}{\beta_1})^{\gamma_1 - 1} e^{(-\frac{x-\mu_1}{\beta_1})}}{\beta_1\,\Gamma(\gamma_1)} \tag{6}$$

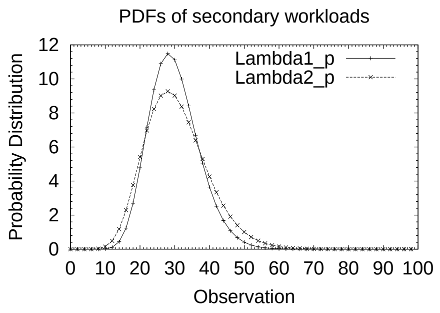


Fig. 21. PDF for Filtered Workloads λ'_1 and λ'_2

and

$$f'_2(x) = \frac{(\frac{x-\mu_2}{\beta_2})^{\gamma_2 - 1} e^{(-\frac{x-\mu_2}{\beta_2})}}{\beta_2 \, \Gamma(\gamma_2)} \tag{7}$$

where x denotes the arrival count per interval and Γ is the Gamma function.

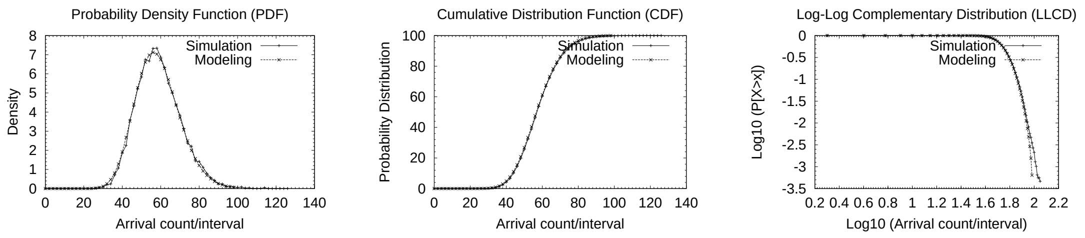
Let M_1, M_2, D_1 and D_2 denote the mean and standard deviation, respectively, of the request arrival count process in a filtered workload (λ'_1 or λ'_2). As an example, we consider an 8 MB cache size at each child proxy, each using the GDS cache replacement policy. From the statistical analyses, we obtain $M_1 = 29.885$, $D_1 = 7.115$ for λ'_1 and $M_2 = 30.570$, $D_2 = 8.928$ for λ'_2.

According to Equation 4 and Equation 5, the Gamma parameters can be estimated as $\hat{\gamma}_1 = 17.642$, $\hat{\beta}_1 = 1.694$ and $\hat{\gamma}_2 = 11.724$, $\hat{\beta}_2 = 2.607$. The Probability Density Functions (PDF) of λ'_1 and λ'_2 are plotted in Figure 21.

Assuming that λ'_1 and λ'_2 are independent, the cumulative request count in the ith time interval for their superposition is governed by:

$$P_{\lambda_3}\{X = k\} = \sum_{i=0}^{k} [P_{\lambda'_1}\{X = i\} \times P_{\lambda'_2}\{X = (k - i)\}] \tag{8}$$

Figure 22 illustrates the characteristics of the request arrival count distribution for the aggregate workload given by the superposition of two request arrival processes. The simulation results are also shown for model validation. Figure 22 shows that the Gamma distribution provides a very good visual fit of the distribution, for both the body and the tail of the distribution. This result demonstrates that a Gamma distribution provides a simple, flexible, and relatively robust means of characterizing the aggregate Web request arrival count distribution. However, the parameters for the Gamma distribution depend upon the input Web workload characteristics and the cache parameters used.



(a) PDF (b) CDF (c) LLCD

Fig. 22. Modeling of Aggregate Workload λ_3

This model can be used to estimate traffic characteristics in hierarchical Web caching architectures, given the input workload characteristics, and the cache configuration parameters.

8 Summary and Conclusions

This paper uses trace-driven simulation to study the structural characteristics of Web workloads in Web proxy caching hierarchies. The simulation experiments quantify the filter effects of a Web cache on the request arrival process, for both empirical and synthetically-generated aggregate Web client workloads. The paper focuses on time-domain analysis of cache filter effects. That is, it focuses on the statistical characteristics of the request arrival count process as transformed by the cache.

Our simulation results demonstrate that the Web proxy cache reduces both the mean arrival rate and the peak arrival rate for Web traffic workloads, but has relatively little impact on the variance. The variance-to-mean ratio of the filtered request arrival process typically increases, depending on the characteristics of the input arrival process and the cache configuration. For a self-similar request arrival process, the results show that the filtered request arrival process remains self-similar, with the same Hurst parameter, though with reduced mean. The study also demonstrates that the superposition of Web request streams from multiple child caches in a Web proxy caching hierarchy remains bursty. Lastly, we find that the Gamma distribution provides a flexible and robust model for characterizing the request arrival process in hierarchical Web caching architectures. However, the parameters for the Gamma distribution are highly dependent upon the cache size and the Web workload characteristics.

Acknowledgments

Financial support for this research was provided by iCORE (Informatics Circle of Research Excellence) and by the Natural Sciences and Engineering Research Council (NSERC) of Canada. The authors are grateful to Greg Oster at the University of Saskatchewan for making traces from the U of S Web proxy available. The authors thank the anonymous reviewers who provided constructive feedback on earlier pieces of this work, appearing at SPECTS [4] and at MASCOTS [5], and the *Performance Evaluation* reviewers who helped streamline the presentation of the paper.

References

[1] R. Addie, M. Zukerman and T. Neame, "Fractal Traffic: Measurements, Modeling, and Performance Evaluation", *Proceedings of IEEE INFOCOM*, Vol. 3, pp. 977-984, April 1995.

[2] V. Almeida, M. Cesario, R. Fonseca, W. Meira Jr., and C. Murta, "Analysing the Behavior of a Proxy Server in Light of Regional and Cultural Issues", *Proceedings of the Third International WWW Caching Workshop*, Manchester, England, June 1998.

[3] M. Arlitt and C. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications", *IEEE/ACM Transactions on Networking*, Vol. 5, No. 5, pp. 631-645, October 1997.

[4] G. Bai and C. Williamson, "Time-Domain Analysis of Web Cache Filter Effects", *Proceedings of SCS Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, San Diego, CA, pp. 195-205, July 2002.

[5] G. Bai and C. Williamson, "Workload Characterization in Web Caching Hierarchies", *Proceedings of IEEE MASCOTS*, Fort Worth, TX, pp. 13-22, October 2002.

[6] R. Balakrishnan and C. Williamson, "The *synTraff* Suite of Traffic Modeling Toolkits", *Proceedings of IEEE MASCOTS*, San Francisco, CA, pp. 333-340, August 2000.

[7] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in Web Client Access Patterns: Characteristics and Caching Implications", *World Wide Web*, Vol. 2, No. 1, pp. 15-28, January 1999.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications", *Proceedings of IEEE INFOCOM*, New York, NY, pp. 126-134, March 1999.

[9] M. Busari and C. Williamson, “On the Sensitivity of Web Proxy Cache Performance to Workload Characteristics”, *Proceedings of IEEE INFOCOM*, pp. 1225-1234, Anchorage, AL, April 2001.

[10] M. Busari and C. Williamson, “Simulation Evaluation of a Heterogeneous Web Caching Hierarchy”, *Proceedings of MASCOTS'2001*, pp. 379-388, Cincinnati, OH, August 2001.

[11] M. Busari and C. Williamson, “ProWGen: A Synthetic Workload Generation Tool for Simulation Evaluation of Web Proxy Caches”, *Computer Networks*, Vol. 38, No. 6, pp. 779-794, June 2002.

[12] H. Che, Z. Wang, and Y. Tung, “Analysis and Design of Hierarchical Web Caching Systems”, *Proceedings of IEEE INFOCOMM*, pp. 1416-1424, Anchorage, AL, April 2001.

[13] M. Crovella and A. Bestavros, “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes”, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 835-846, December 1997.

[14] M. Crovella and A. Bestavros, “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes” *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 835-846, December 1997.

[15] R. Doyle, J. Chase, S. Gadde, and A. Vahdat, “The Trickle-Down Effect: Web Caching and Server Request Distribution”, *Proceedings of the Web Caching and Content Delivery Workshop*, Boston, MA, June 2001.

[16] L Fan, P. Cao, J. Almeida and A. Broder, “Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol”, *IEEE/ACM Transactions on Networking* Vol. 8, No. 3, pp. 281-293, June 2000.

[17] A. Feldmann, R. Cáceres, F. Douglis, G. Glass and M. Rabinovich, “Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments”, *Proceedings of IEEE INFOCOM*, pp. 107-116, April 1999.

[18] R. Fonseca, V. Almeida, M. Crovella, and B. Abrahao, “On the Intrinsic Locality Properties of Web Reference Streams”, *Proceedings of IEEE INFOCOM*, San Francisco, CA, April 2003.

[19] M. Franklin, M. Carey, and M. Livny, “Global Memory Management for Client-Server DBMS Architectures”, *Proceedings of the 19th International Conference on Very Large Databases* (VLDB), August 1992.

[20] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, “On the Self-Similar Nature of Ethernet Traffic (Extended Version)”, *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, pp. 1-15, February 1994.

[21] A. Mahanti, C. Williamson, and D. Eager, “Traffic Analysis of a Web Proxy Caching Hierarchy”, *IEEE Network*, Vol. 14, No. 3, pp. 16-23, May/June 2000.

[22] N. Markatchev and C. Williamson, "WebTraff: A GUI for Web Proxy Cache Workload Modeling and Analysis", *Proceedings of IEEE MASCOTS*, Fort Worth, TX, pp. 356-363, October 2002.

[23] N. Markatchev and C. Williamson, "Network-Level Impacts on User-Level Web Performance", *Proceedings of SCS Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, Montreal, PQ, pp. 637-646, July 2003.

[24] D. Menascé and V. Almeida, *Capacity Planning for Web Services: Metrics, Models, and Methods*, Prentice-Hall, 2002.

[25] V. Paxson and S. Floyd, "Wide Area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Transactions on Networking*, Vol. 3, No. 3, pp. 226-244, 1995.

[26] A. Patel and C. Williamson, "Effective Bandwidth of Self-Similar Traffic Sources: Theoretical and Simulation Results", *Proceedings of the IASTED Conference on Applied Modeling and Simulation*, Banff, AB, pp. 298-302, July 1997.

[27] C. Roadknight, I. Marshall, and D. Vearer, "File Popularity Characterization", *Proceedings of the Second Workshop on Internet Server Performance (WISP)*, Atlanta, GA, May 1999.

[28] D. Weikle, S. McKee, and W. Wulf, "Caches as Filters: A New Approach to Cache Analysis", *Proceedings of MASCOTS'98*, Montreal, PQ, pp. 2-12, July 1998.

[29] C. Williamson, "On Filter Effects in Web Caching Hierarchies", *ACM Transactions on Internet Technology*, Vol. 2, No. 1, pp. 1-31, February 2002.

[30] D. Willick, D. Eager, and R. Bunt, "Disk Cache Replacement Policies for Network Fileservers", *Proceedings of ICDCS*, Pittsburgh, PA, 1993.

Guangwei Bai received his B.Eng. and M.Eng. from Xi'an Jiaotong University, China, both in computer engineering. He received his Ph.D. in Computer Science from the University of Hamburg, Germany. He worked at GMD (German National Research Center for Information Technology) as a Research Scientist from 1999 to 2001. Since 2001, he has been working at the University of Calgary, Canada, as a Research Associate. His research interests are in the measurement, modeling, workload characterization, and performance analysis of Internet, wireless, and multimedia communication systems.

Carey Williamson is an iCORE Professor in the Department of Computer Science at the University of Calgary, specializing in *Broadband Wireless Networks, Protocols, Applications, and Performance.* He holds a B.Sc.(Honours) in Computer Science from the University of Saskatchewan, and a Ph.D. in Computer Science from Stanford University. His research interests include Internet protocols, wireless networks, network traffic measurement, network simulation, and Web server performance.