

Analysis of Performance Impact of Drill-Down Techniques for Web Traffic Models

Cathy H. Xia, Zhen Liu, Mark S. Squillante, Li Zhang, Naceur Malouch
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598, USA
{cathyx, zhenl, mss, zhangli}@us.ibm.com
Naceur.Malouch@sophia.inria.fr

The performance of Web sites continues to be an important research topic. Such studies are invariably based on the access logs from the servers comprising the Web site. A problem with existing access logs is the coarse granularity of the timestamps, e.g., arrival times. In this study we demonstrate and quantify the significant differences in performance obtained under diverse assumptions about the arrival process of user requests derived from the access logs, where the corresponding user response times can differ by more than an order of magnitude. This motivates the need for a general methodology to construct accurate representations of the actual arrival process of user requests from existing coarse-grained access-log data. Our analysis of the access logs from representative commercial Web sites illustrates self-similar behavior of the arrival process, and we exploit the properties of these self-similar processes as a theoretical foundation for constructing the arrival process at finer time scales. The advantage of our approach is that it maintains consistency between the properties of the arrival processes at both coarser and finer time scales. In addition, our analysis of the request size distribution from commercial Web sites demonstrates a subexponential, but not heavy-tail (power-law) distribution. Through simulations, we investigate the impact of these different traffic models on the user request response times.

1. Introduction

With the rapid advances in Internet technology, e-commerce is becoming a mature business strategy. The concept of Quality of Service (QoS) is working its way to the front lines of e-business commitments and requirements as it plays an important role in Internet applications, services and pricing negotiations. One needs to have a fundamental understanding of the key characteristics of the workload patterns in production Web sites and a fundamental understanding of the impact of such workload patterns on Web server performance as well as the server capacity required to guarantee a certain level of QoS, e.g., user request response time. Our primary focus in this study is to investigate these important research issues related to such performance metrics in the context of a real commercial Web server environment.

A key aspect of our study must therefore concern the arrival patterns of incoming user requests and the resource requirements for serving these requests, as these represent two of the most important components of commercial Web site workloads. Such workload characterizations have been consistently based on the access logs from a specific Web site of interest [1,5,11,22,15], possibly together with additional measurement data. These logs contain information on all of the HTTP requests made to the server, including the client IP address, the time of the request, and the size of the requested object. The user request times are commonly provided at the granularity of a second. Typically there are tens or even hundreds of requests within a second for the commercial Web sites considered in our study.

This lack of finer granularity in the user request arrival times leaves a potentially important gap in

our understanding of the user request interarrival process. In particular, accurate modeling of this arrival process can be crucial for the response time characterization of these user requests. We therefore examine the impact of this lack of more detailed arrival information on the performance of the Web site. Our investigation shows that different assumptions on how the user requests arrive within each one second interval can have a significant impact on various QoS measures, which are especially important for commercial Web sites. This includes more than an order of magnitude difference in the mean response time for user requests. Given these results, our study identifies a critical need for accurately estimating the manner in which user requests arrive within each one second interval of the Web server access log.

A primary focus of this study is thus devoted to developing a methodology for drilling down to finer time scales in order to properly spread out the per-second batch process of requests, and to evaluate the corresponding performance impact. Specifically, we propose a drill-down methodology based on an analysis of the user request arrival process at coarser time scales to accurately construct the request arrival process at finer time scales. Our analysis of the access logs from a representative commercial Web site illustrates self-similar (SS) behavior of the request arrival processes during the peak traffic periods, which typically last for relatively long periods of time, i.e., on the order of several hours. The properties of the corresponding SS processes then provide a theoretical foundation for accurately constructing the arrival process at finer time scales, and thus we exploit these properties to construct the corresponding interarrival processes of user requests. The advantage of our approach is that it maintains consistency between the statistical properties of the arrival processes at both coarser and finer time scales.

Our study also considers the other important aspect of commercial Web site workloads, namely the resource requirements for serving user requests. In particular, we analyzed the distribution function (d.f.) of the sizes of objects requested by users in the commercial Web site of interest. Our results show that the corresponding d.f. does not fall into the class of heavy-tail (HT) d.f.s with a power law, as shown in a number of studies of data from different Web environments [5,3]. Instead, a subexponential Weibull d.f. appears to provide a good fit, and thus our results are more consistent with those in [22,6,12].

It is important to point out that our sole focus in this study is on the higher-level user request traffic to the Web site of interest. This is in contrast to the lower-level packet traffic that has been investigated in a considerable amount of previous work; e.g., see [14,23,18,21,8,10] and the references cited therein. While some previous studies have considered the user request traffic for certain types of Web servers (e.g., see [1,5,3,11,22,15]), these studies have not considered the user request traffic found at real, production-level, commercial Web sites. Moreover, to the best of our knowledge, our study is the first to demonstrate the importance of understanding arrival patterns of user requests at finer time scales than one second, and to develop a drill-down methodology to obtain an accurate representation of the user request interarrival process at such finer time scales.

Given the results of this analysis based on our drill-down methodology, we then investigate through simulations the impact of the different traffic models on the user request response times. Since a primary focus of this paper is on the stochastic arrival and service processes for the Web server of interest, we use a $G/G/1$ queue under the processor sharing discipline to model the Web server as a first-order approximation. Our study provides insights on the performance impact of both the long-range dependent (LRD) arrival process and subexponential (SE) service times in processor sharing (PS) queues, and illuminates the different dominating components that influence server performance under various conditions.

2. Request Arrival Processes

2.1. Measurement Data

The Web traffic used in this study is obtained from a commercial Web site from the *retail industry* whose identity will remain anonymous for obvious confidentiality restrictions. This commercial Web site provides fairly typical services for the retail industry, such as browsing and buying of various types of goods and services. We have obtained the access log files of each of the servers from this commercial

Web site. The logs of over several weeks are used as the basis for our study. These logs contain information on all of the HTTP requests made to an individual server, including the client IP address, the time of the request, and the size of the page or object.

The traces used in our study represent more than eight weeks of Web access logs from two separate months in 2001, where different aspects of the daily traffic patterns are recorded; this includes the per request arrival times and object size. The time-scale of a second and size-scale of a byte are the granularity used in the server logs available to us, which is the predominant case in commercial Web sites of recent years. Given the main objectives of our study, we identify and focus on sufficiently long stationary intervals of peak traffic periods found in each of the access logs of interest.

Our analyses of such stationary intervals of the access logs from various server nodes of the commercial Web site of interest yield very similar arrival and service time processes, from which we selected representative data sets of a server to be used as the basis for our study. These data sets consist of peak traffic periods whose lengths are on the order of several hours and consist of over 2 terabytes of data being transmitted by more than 500,000 user requests.

2.2. Batch vs. Continuous Arrival Processes

Studies on the performance of Web sites are invariably based on the access logs from the servers comprising the Web site. A problem with existing access logs is the coarse granularity of the time-scale, e.g., arrival times. The fact that most commercial Web server access logs record time at the granularity of seconds, typically resulting in tens or hundreds of requests per second, can make it difficult to address performance issues at the per-request level, since the specific interarrival times are missing.

There are several heuristics we can apply to resolve this issue. First and simplest, we can assume that requests arrive in batches every second as recorded in the logs, which would obviously provide a pessimistic estimation of the per-request performance that is clearly an upper bound. We will refer to this arrival pattern as *Batch*. Another approach is to assume that the interarrival times for the requests within the same second are equally spaced in a deterministic manner. This is optimistic as it ignores the variability of the interarrival times, and thus we shall refer to it as *DBatch*. A third method is to assume that the arrival times within the second are uniformly distributed, which is equivalent to assuming Poisson arrivals within the second. We will refer to this arrival process as *UBatch*. An obvious drawback of this *UBatch* approach is that it assumes independence between arrivals within each second, which has been challenged by examinations of network traffic in [14,19].

These different assumptions on the arrival patterns can lead to very different conclusions with respect to the resulting performance impact. For example, the performance differences between *Batch* and *DBatch* arrivals could easily exceed an order of magnitude, as demonstrated by experiments with the available commercial Web site access logs. The extremely large performance gaps strongly motivate the need for a better understanding of the user request arrival times at much finer time-scales than a second. Note also that all three of the above assumptions on the individual request arrival patterns disregard the statistical information that is contained in the data. Since the traffic statistics (at the observable per-second level) can actually provide us with guidelines at finer time scales, we are motivated to further capture the statistical characteristics of the arrival process, which is presented next.

2.3. Evidence of Self-similarity

The most important aspects of SS traffic are that the traffic has observable bursts on all time scales and that it exhibits LRD. The degree of SS can be expressed in a single parameter, i.e., the *Hurst* parameter $H \in [\frac{1}{2}, 1)$. Examinations of network traffic [14,19] and World Wide Web traffic [1,5] all provide evidence of SS, which have greatly challenged the commonly assumed Poisson models. For detailed discussions of SS time series and the corresponding statistical tests, please refer to [4,23].

We use a number of different methods to test for SS of the arrival processes. These include heuristic approaches based on the correlation plot and the variance plot, parametric statistical approaches based

on the Maximum Likelihood Estimation (MLE) methods using Whittle’s estimator [4], and wavelet-based approaches recently developed by Abry and Veitch [2]. Though the wavelet-based technique is known to be robust and efficient in investigating scaling properties for large data sets, we also rely on the model-based MLE methods to gain confidence in choosing a specific traffic model.

Figure 1 illustrates the results of such SS tests on a representative trace of the number of requests per second over a five-hour peak-traffic interval. In Figure 1, the two leftmost plots respectively show the autocorrelation plot and the variance plot, both in log scale. The plot in the upper rightmost corner shows, in the spectral domain, the periodogram as a function of the frequency in log scale, where two different methods were used to estimate the Hurst parameter; namely, the least square regression (LSQ), and the MLE method based on the fractional Gaussian noise (FGN) model. The p-value shows the goodness of fit under the given model. Thus, the higher the p-value, the more confidence there is in the chosen model as a good candidate. Finally, the lower rightmost plot shows the Logscale Diagram under the second order wavelet analysis [2]. Note that if the trace is exactly SS, the Logscale Diagram will show a linear relationship across all scales.

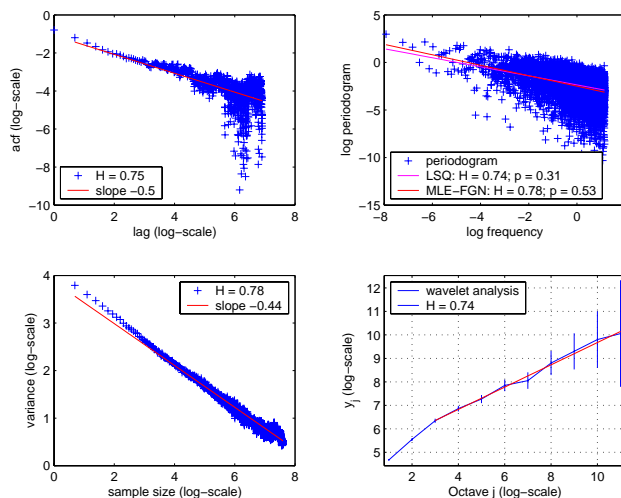


Figure 1. Long-range dependency tests on the REQ/sec trace show evidence of LRD.

Observe that almost all tests yield similar Hurst parameter estimations in the range of $H \approx 0.74 - 0.8$. The clear linear relationship across almost all scales in the Logscale Diagram (of the wavelet analysis) suggests that the request-level trace is close to exact SS. The MLE estimator, based on the FGN model, also provides the highest p-value indicating that the FGN model is a good fit. These results provide considerable statistical evidence which strongly suggests that the peak-traffic traces exhibit behavior that is consistent with SS traffic models.

3. Drill-Down Methodology

In this section, we present a drill-down method that makes it possible to regenerate the arrival process at much finer time scales based on the statistical characteristics of the observed input traffic at the time granularity of the server access log. This then provides statistically more accurate interarrival times than the *Batch*, *DBatch* and *UBatch* schemes for our performance study.

3.1. Drill-Down Methodology Based on Self-Similarity

It is known that for an exactly SS process, the degree of irregularity is the same across all points in time and under all scales. This then provides us statistical guidelines for the process at finer time scales. Given the strong evidence of SS of the Web server data from Section 2, we use fractional Brownian motion (FBM) to model the input traffic. It is important to note that different drill-down techniques,

within the context of our general methodology, can be analogously developed based on models other than FBM. For now we leave this as a future research topic.

An FBM process $A(t)$ with mean drift rate λ and variance coefficient a , is defined as [4]

$$A(t) = \lambda t + \sqrt{\lambda a} Z(t), \quad t \in (-\infty, \infty), \quad (1)$$

where $Z(\cdot)$ is a normalized FBM with mean 0, variance 1, and Hurst parameter $H \in [\frac{1}{2}, 1)$; it also has a covariance function $\Gamma(s, t) = E[Z(s)Z(t)] = \frac{1}{2} (s^{2H} + t^{2H} - |s - t|^{2H})$. The increment process $X_n = A(n) - A(n - 1)$ ($n = 1, 2, \dots$) is then called fractional Gaussian noise (FGN).

Suppose that the arrival sequence $\{X_n\}$, where X_n denotes the number of arrivals at time n , is an FGN process. Let $A(t)$ be the cumulative number of arrivals in $[0, t)$. We can then model the arrival process $A(t)$ as an FBM process given by (1). Let $A_s(t)$ be the corresponding process at a finer time scale s ; for example, $s = 1/1000$ represents the input traffic at the millisecond level. Based on (1) and the fact that $Z(st) \stackrel{d}{=} s^H Z(t)$, for every $s > 0$, where $\stackrel{d}{=}$ means equal in distribution, we then have

$$A_s(t) = A(st) = \lambda st + \sqrt{\lambda a} Z(st) \stackrel{d}{=} \lambda_s t + \sqrt{\lambda_s a_s} Z(t), \quad (2)$$

where the mean drift rate is $\lambda_s = \lambda s$, and the variance coefficient is equal to $a_s = a s^{2H-1}$. Equation (2) provides the fundamental properties of the cumulative arrival process (with respect to the number of requests) at a finer time scale s . This then motivates the following drill-down method.

The first step consists of characterizing the SS of the empirical Web server data (at the second level). We then apply (2) to re-generate the traffic at a given finer time scale s . The traffic generation is based on the FFT-Algorithm proposed in [20]. In particular, a sequence of FGN increments ξ_n , $n = 1, 2, \dots$, is generated such that $\xi_n = A_s(n) - A_s(n - 1)$ with $A_s(0) = 0$.

Given that the number of requests per second is on the order of tens and at most hundreds, then when viewing the arrival process at a much finer time scale s , say every two milliseconds (we find that scale $s = 1/500$ is sufficiently small), we have a sequence of binary numbers; namely, either there is a request at that millisecond or there is not. Since the generated sequence $\{\xi_n\}$ is in real numbers, in order to have a binary arrival sequence $\{\xi'_n\}$, we apply a rounding scheme so that ξ'_n is set to 1 whenever the sum $\sum_{i=1}^n \xi_i$ is close enough to an integer number. The rounding scheme basically records a new arrival each time that the cumulative $A_s(\cdot)$ is close to an integer number. If we denote by $A'(n)$ the cumulative sum of the binary sequence $\{\xi'_n\}$, then the above rounding scheme keeps A' close to $A_s(\cdot)$ through all arrival times. Consider the binary sequence ξ'_n , $n = 1, 2, \dots$ as a record of the occurrences of arrivals at the finer scale s . We then have reconstructed the individual request arrival times at the finer scale s based on the SS of the original process. We will denote this arrival sequence from our drill-down method based on self-similarity as *DD-SS*.

Our LRD tests on the newly generated traffic after the rounding scheme and aggregation back to the per second level, show that the dependence structure is in nice agreement with the original data. Similar tests were carried out on the generated arrival traffic at finer time scales, which also show SS as expected. All of this suggest that our drill-down method works well.

3.2. Comparison with Other Arrival Patterns

In order to compare our drill-down method with other heuristic methods, we apply to the original trace four different schemes, namely *DBatch*, *UBatch*, *Poisson* and *DD-SS*, where the *DD-SS* arrival process is obtained by our drill-down method where the underlying FBM process has (based on estimations from the original trace) Hurst parameter $H = 0.78$, arrival rate $\lambda = 22.36$ -requests/sec, and variance coefficient $\lambda a = 61.57$. The *DBatch* and *UBatch* schemes are described in Section 2. Note that the resulting four different arrival processes have the same arrival rate λ . We next explore the difference in both the statistical characteristics of the interarrival times, and the corresponding performance impact.

1) *Interarrival Times*: Denote by t_i the time epoch of the i -th arrival of the FBM process $A(\cdot)$, that is, $A(t_i) = i$, for $i = 1, 2, \dots$. The interarrival times are $T_i = t_i - t_{i-1}$, where it is assumed that $t_0 = 0$. Note that $\mathbf{P}[T_1 < t] = \mathbf{P}[A(t) > 1] = \mathbf{P}\left[\lambda t + \sqrt{\lambda a} Z(t) > 1\right] = \Phi\left(\frac{1 - \lambda t}{\sqrt{\lambda a t^H}}\right)$, where $\Phi(x) = \mathbf{P}[Z(1) > x]$ denotes the tail d.f. of standard normal variables. For $i > 1$, T_i can be considered as the forward recurrence time of T_1 , therefore its tail d.f. is simply the integrated tail of T_1 . When $H = 1/2$, this integrated d.f. is Inverse Gaussian.

The empirical probability density functions of the interarrival times under the four different methods are shown in the leftmost plot of Figure 2. We see that the interarrival times under *UBatch* and *Poisson* both have exponential marginal d.f.s, which are clearly different from the Gaussian and Inverse Gaussian type marginal d.f.s exhibited under the *DBatch* and *DD-SS* schemes. The autocorrelation functions of the interarrival times are shown in the two rightmost plots of Figure 2, where the middle one shows the short-range autocorrelation function (a.c.f.) from lag 0 to lag 5, and the right one shows the long-range behavior from lag 100 to lag 900. We see that much higher correlations are introduced under the *DBatch* scheme than under other schemes. Though the *UBatch* scheme exhibits similar long-range auto-correlations as the *DD-SS* scheme, its short-term correlations are lower than that of the *DD-SS* scheme, since it assumes independence of the interarrival times within each second. Furthermore, this difference in a.c.f. between *UBatch* and *DD-SS* scheme increases as the Hurst parameter increases (i.e., stronger dependence structure), or as the average number of requests per second increases.

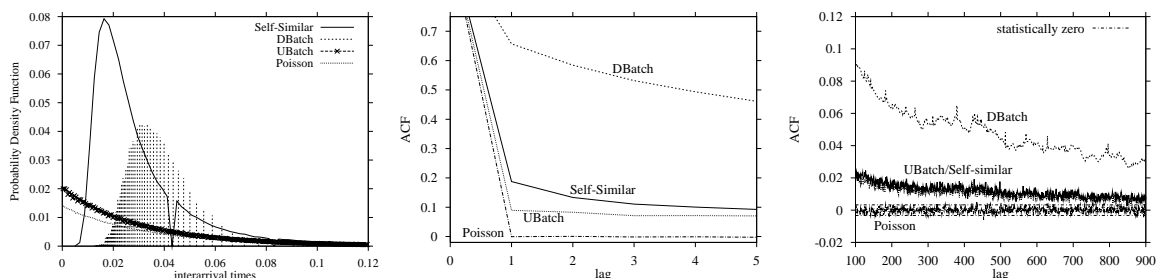


Figure 2. **Left**: Marginal d.f.s of the interarrival times under different arrival patterns. **Middle**: a.c.f. of the interarrival times from lag 0-5. **Right**: a.c.f. of the interarrival times from lag 100-900.

2) *Impact on Response Times*: Simulations are carried out in order to further investigate the performance impact under different arrival assumptions. We feed different arrival processes as input to a PS queue to simulate per-request response times at the Web server of interest. Specifically, we consider five different arrival schemes, *Batch*, *DBatch*, *UBatch*, *Poisson*, and the *DD-SS* arrival using the drill-down scheme, all having the same arrival rate λ . The service times are constant at rate μ , where μ is chosen so that the traffic intensity is $\rho = 0.8$. We assume constant service times at this point so that we can focus on the performance impact purely due to the variability and dependence structure of the arrival process. The impact on performance of different service time d.f.s will be addressed in subsequent sections.

The empirical tail d.f.s of the simulated per-request response times under different arrival patterns are shown in the leftmost plot of Figure 3. Observe that the performance under *Poisson* arrivals is far better than the other schemes for large x , indicating that the dependence structure of the arrival process has a dramatic impact on performance. On the other hand, the *Batch* scheme dramatically overestimates the performance. Almost all requests under the *Batch* scheme suffer large delays. Though different within the second interval, the *DD-SS*, *UBatch*, and *DBatch* schemes all seem to have maintained the LRD to some degree. In addition, the *DBatch* scheme consistently provides lower tail probabilities than *DD-SS* and *UBatch*, with more than 50% of requests experiencing almost no delay, since it is over-optimistic in estimating the variability of interarrival times. The tail behaviors of *DD-SS* and *UBatch* are the closest.

However, the performance difference between the *UBatch* and *DD-SS* schemes increases as the average number of requests within a batch increases. In the rightmost plot of Figure 3, we show that if

the available arrival process trace is aggregated at the minute level, or, in other words, when the average number of requests within a second is very large, the *UBatch* scheme becomes closer to the *Poisson* process, and the performance exhibits very different tail asymptotics than that under the SS arrival processes; whereas the *DD-SS* scheme maintains consistency in representing the dependence structure at different arrival rates and across different time scales. Note that in this case, the number of jobs within a batch is so large that the *DBatch* scheme is able to equally space out almost all of the requests, and as a result, almost all requests experience no delay; the large batch sizes make arrivals under the *Batch* scheme suffer further, and the user request response times under the *Batch* scheme become more than an order of magnitude larger than the rest, which cannot fit in the same picture without losing the capability of distinguishing it from the others.

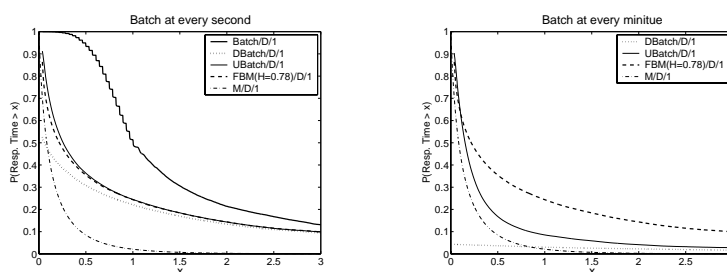


Figure 3. Impact on response times. **Left:** The available batched arrival process is at the second level. **Right:** The available batched arrival process is at the minute level.

4. Joint Performance Impact by Arrival Patterns and Service Time Distributions

We have shown that the performance impact of different arrival patterns can be significant. It is well-known that non-conventional service time d.f.s can also have a significant impact on system performance [17]. Therefore, it is important to address the joint impact on Web server performance of both the arrival time and service time d.f.s.

4.1. Service Time Distribution

In order to capture the service time d.f., we analyze the empirical d.f. of the size of the objects requested by users in the commercial Web site of interest. Many previous studies [11,5,3] have used user request sizes as a surrogate for service times. In this paper, we shall assume, as similarly observed in [11], that the service times are proportional to the request sizes.

Our careful statistical analyses show that the corresponding d.f. does not fall into the class of HT d.f.s with a power law, as shown in a number of studies of data from different Web environments [5,3]. Instead, a Weibull d.f. (with $\beta \approx 0.6$) appears to provide a good fit for the request size d.f., suggesting that it is SE but not HT. Thus our results are more consistent with those in [22,6,12,9].

4.2. Simulation Results: Performance under LRD Arrivals and Subexponential Service Times

As we mentioned previously, the Web site under consideration consists of multiple server nodes. The arrival processes to these servers have similar statistical properties. We thus only need to focus on one of the Web servers. Also, these servers operate under a time-sharing operating system. We therefore use an *FBM/G/1 queue under the PS discipline* to model the Web server as a first-order approximation. Here the arrival process is SS and modeled as an FBM process; the service times are independent and identically distributed (i.i.d.) random variables (r.v.s) with marginal d.f. F (independent of the arrival process).

Simulations are carried out to investigate the performance impact of both the LRD arrival process and SE service times. The input process of the FBM/G/1 queue is generated using the drill-down method of Section 3 based on the FBM model with $H = 0.78$, $\lambda = 22.36$ -requests/sec and $\lambda a = 61.57$. We then simulate the system with this synthetic FBM input process and explore the performance measures under

different service time d.f.s. In addition to the measured service times, we also use other random service times generated under different d.f. assumptions. The simulated response time tail d.f.s are then plotted in comparison with that of an M/G/1/PS system (with the same service times but assuming Poisson inputs), an FBM/M/1 system with FBM inputs but exponential service times, and an M/M/1/PS queue with Poisson inputs and exponential service times. In order to carry out a fair comparison, we fix the traffic intensity at $\rho = 0.8$, the arrival rate at $\lambda = 22.36$ -requests/sec, and the service rate at $\mu = \lambda/\rho$.

Note that the performance of an M/G/1/PS queueing system shows the pure impact of the service time d.f.s; whereas the performance of an FBM/M/1/PS queueing system illustrates the pure impact by the LRD of the arrival process. The tail d.f. of the response time in an M/G/1/PS system was studied in [13], in which it was shown that

$$\mathbf{P}[\text{Resp.Time} > x] = \mathbf{P}[S > (1 - \rho)x] (1 + o(1)), \quad \text{as } x \rightarrow \infty, \quad (3)$$

when the d.f. of the job size S belongs to a class of SE d.f.s with tails heavier than $e^{\sqrt{x}}$. The closest results related to a FBM/M/1/PS system are probably [7,16], in which it was shown that the stationary backlog process has an asymptotic Weibull tail marginal d.f. with shape parameter $2 - 2H$, where H is the Hurst parameter of the FBM process.

Figure 4 shows the simulated response time tail d.f.s (with y axis in log-scale), when the service times follow the Weibull¹ d.f. with $\beta = 0.8$ and $\beta = 0.2$, respectively, and the corresponding parameter α is chosen so that the mean service time is μ^{-1} . Note that in both cases the service times are SE, however, Weibull(0.8) has a lighter tail than Weibull(2-2H) with $H = 0.78$, while Weibull(0.2) has a heavier tail. Observe that under Weibull(0.8) service times, the tail d.f. of the response times under FBM inputs is very close to the tail d.f. of the FBM/M/1 queue, suggesting that the dependence structure dominates the performance. However, when the service times are Weibull(0.2), the performance deviates greatly from that of FBM/M/1 queues, but stays much closer to that of the M/Weibull(0.2)/1 queues, suggesting that the service times instead dominate the performance when the service time tail d.f. is heavier than Weibull(2-2H). Note that with the observed user request size from logs, the service time d.f. fits best to a Weibull d.f. with shape parameter $\beta = 0.6$. In this case, the corresponding performance impact due to the service time d.f. is again small and the dependence structure dominates the performance. Due to space limitations, the simulation results are omitted since it is similar to the leftmost plot in Figure 4.

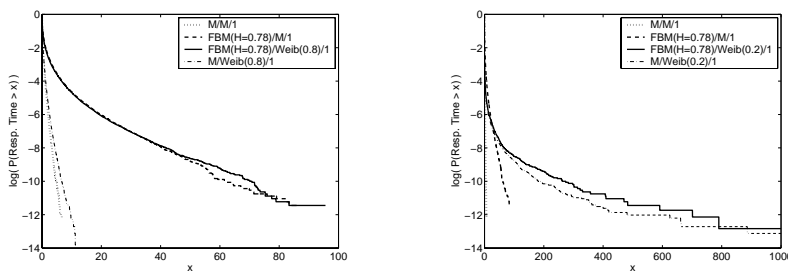


Figure 4. Response time tail d.f. comparisons. SS arrival $H=0.78$. **Left:** Under Weibull(0.8) service times; **Right:** Under Weibull(0.2) service times.

The next set of simulations were carried out when the service times follow the Pareto² d.f. with $\alpha = 4.0$ and $\alpha = 1.5$, respectively, and the corresponding parameter β is chosen so that the mean service time is μ^{-1} . Note that Pareto(4.0) has finite variance, while Pareto(1.5) is HT with infinite variance. In

¹The Weibull d.f. with scale parameter α and shape parameter β is defined as $F(x) = 1 - e^{-ax^\beta}$, $x > 0$. The mean of a Weibull r.v. with parameter (α, β) is $\alpha^{-1/\beta} \Gamma(1 + 1/\beta)$, where $\Gamma(y) = \int_0^\infty e^{-x} x^{y-1} dx$.

²A d.f. F is Pareto with shape parameter α and scale parameter β if $F(x) = 1 - (x/\beta)^{-\alpha}$, $x \geq \beta > 0, \alpha > 0$. The mean of a Pareto r.v. with parameter (α, β) is $\frac{\beta\alpha}{\alpha-1}$.

order to have a clear picture of the performance impact of the dependent arrival process, we consider separately two SS arrival processes with Hurst parameter $H = 0.6$ and $H = 0.78$ respectively.

The top two plots in Figure 5 show the response time tail d.f.s (in log-log scale) when the arrival process is SS with $H = 0.6$ and the service times follow a Pareto d.f. Note that when the arrival process is Poisson, based on (3), the tail d.f. of the corresponding M/Pareto(α)/1/PS queue is asymptotically Pareto(α), which should asymptotically look like a straight line in the log-log plot. This can be clearly verified in Figure 5. Observe that when the service times follow Pareto(1.5), the performance under SS arrivals clearly becomes very close to that of the M/Pareto(1.5)/1 queues. However, when the service times follow Pareto(4.0), which is not HT, the performance differences among the four systems are not clear. This suggests that when service times are HT, the response time tail behavior is essentially dominated by the tail behavior of the service time d.f.

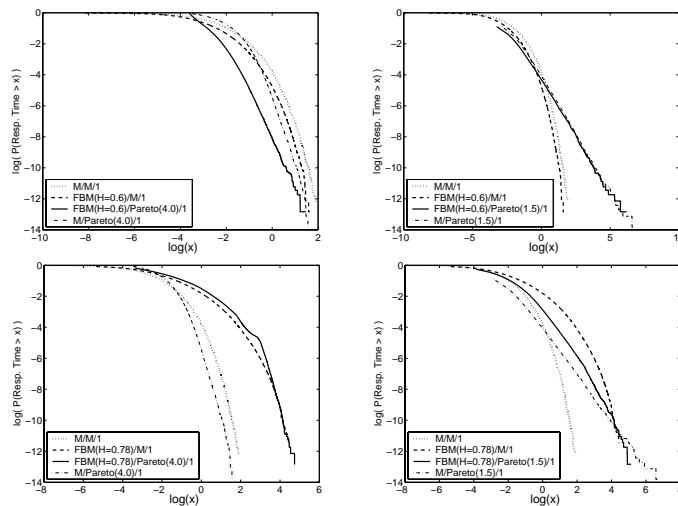


Figure 5. Response time tail d.f. comparisons. **Upper-left:** Under FBM($H=0.6$) inputs and Pareto(4.0) service times; **Upper-right:** Under FBM($H=0.6$) inputs and Pareto(1.5) service times. **Lower-left:** Under FBM($H=0.78$) inputs and Pareto(4.0) service times; **Lower-right:** Under FBM($H=0.78$) inputs and Pareto(1.5) service times.

The bottom two plots in Figure 5 show the simulation results as the arrival process is FBM with Hurst parameter $H = 0.78$ (whose dependence structure is much stronger than that with $H = 0.6$.) Observe that when the service times follow Pareto(4.0) (in the right-center plot), the response times under the FBM input process are almost like the backlog in an FBM/M/1 queue, suggesting that the dependence structure of the arrival process is dominant, while the impact on performance by the service times becomes negligible. However, when the service times follow Pareto(1.5) (in the rightmost plot), the performance under SS arrivals again becomes close to that of the M/Pareto(1.5)/1 queues, though the impact of dependence structure cannot be totally neglected. This suggests that when service times are HT, the impact on performance of the service time d.f. is dominant.

4.3. Summary on the Joint Impact

Based on the results of our analysis and our methodology for constructing accurate arrival processes of user requests, we investigated through simulations the impact of different traffic models on the user request response times. Our simulation results show that, in a PS queueing system, when the arrival process is LRD and the service times are SE, the tail d.f. of user request response times tend to be dominated by the heavier of the two between the Weibull(2-2H) d.f. and the service time tail d.f., where the Weibull(2-2H) shows the performance impact purely by the LRD arrival process.

Our study provides insights on the performance impact by both the LRD arrival process and SE service

times in PS queues, and illuminates the different dominating components that influence server performance under various conditions. Though these observations are only based on simulations, we conjecture that they are asymptotically exact. This is the subject of our on-going investigation.

REFERENCES

1. M.F. Arlitt and C.L. Williamson. Internet Web servers: Workload characterization and performance implications. *IEEE/ACM Trans. on Networking*, 5(5):631-645, Oct. 1997.
2. P. Abry and D. Veitch. Wavelet analysis of long-range dependence traffic, *IEEE Trans. on Information Theory*, 44(1):2-15, 1998.
3. P. Barford and M.E. Corvella. Generating representative Web workloads for network and server performance evaluation. *Proc. of ACM SIGMETRICS Conf.*, pp. 151-160, 1998.
4. J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, New York, 1994.
5. M.E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: Evidence and possible causes. *Proc. of ACM SIGMETRICS Conf.*, May 1996.
6. A. Downey. The structural cause of file size distributions. *Proc. of MASCOTS*, Aug. 2001.
7. N.G. Duffield and N. O’Connell. Large deviation and overflow probabilities for the general single-server queue, with applications. *Math. Proc. Camb. Phil. Soc.*, **118**(1995) 363-375.
8. A. Feldman, A.C. Gilbert, W. Willinger and T.G. Kurtz. The changing nature of network traffic: scaling phenomena. *Computer Communication Review*, vol. 28, no. 2, pp. 5–29, April 1998.
9. A. Feldman and W. Whitt. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. *Performance Evaluations*, 31, 1998.
10. M. Grossglauser and J. Bolot. On the relevance of long range dependence in network traffic. *IEEE/ACM Transactions on Networking*, 1998.
11. A.K. Iyengar, M.S. Squillante, and L. Zhang. Analysis and characterization of large-scale Web server Access Patterns and Performance. *World Wide Web*, **2**(1999) 85-100.
12. C. Jalpa-Villanueva, Z. Liu, N. Niclausse, S. Barbier, Web traffic characterization using mixture of parsimonious distributions. *Proc. of Communication Networks and Distributed Systems Modeling and Simulation Conference*, San Antonio, Texas, Jan. 2002.
13. P. Jelenkovic and P. Momcilovic. Resource sharing with subexponential distributions. *Proc. of IEEE INFOCOM*, June 2002, **3** 1316-1325.
14. W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similarity nature of Ethernet traffic (Extended version). *IEEE/ACM Trans. on Networking* (1994), vol. 2 No.1 1-15.
15. Zhen Liu, Nicolas Niclausse, and Cesar Jalpa-Villanueva. Traffic model and performance evaluation of Web servers. *Performance Evaluation*, Oct. 2001, **46** 77–100.
16. I. Norros. A storage model with self-similar input. *Queueing Systems*, **16**(1994) 387-396.
17. A. Pakes. On the tails of response-time distributions. *J. Appl. Prob.* **12**(1975) 555-564.
18. K. Park, G.T. Kim and M.E. Crovella. On the effect of traffic self-similarity on network performance. *Proc. of SPIE Intl. Conf. on Performance and Control of Network Systems*, Nov., 1997.
19. V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. *Proc. of SIGCOMM’94*.
20. V. Paxson. Fast approximation of self-similar network traffic. Tech. report, *LBL-36750/UC-405*, April 1995.
21. R. H. Riedi and J. Lévy Véhel. Multifractal properties of TCP traffic: a numerical study, *INRIA Research Report* 3129, March 1997.
22. M.S. Squillante, D.D. Yao, and L. Zhang. Web traffic modeling and server performance analysis. *Proc. of IEEE Conference on Decision and Control*, 4432-4439, 1999.
23. W. Willinger, M.S. Taqqu, W.E. Leland, and D.V. Wilson. Self-similarity in high-speed packet traffic: Analysis and modeling of Ethernet traffic measurements. *Statistical Science*, 10(1):67-85, 1995.