Queueing Analysis of Network Traffic: Methodology and Visualization Tools

D. A. Rolls [a,*], G. Michailidis [b], F. Hernandez-Campos [c]

[a] *Dept. of Mathematics & Statistics, University of North Carolina at Wilmington, Wilmington, NC 28403*

[b] *Dept. of Statistics, University of Michigan, Ann Arbor, MI, 48109-1092*

[c] *Dept. of Computer Science, University of North Carolina, Chapel Hill, NC 27599-3175*

Abstract

In this paper we provide a framework for analyzing network traffic traces through trace-driven queueing. We also introduce several queueing metrics together with the associated visualization tools (some novel) that provide insight into the traffic features and facilitate comparisons between traces. Some techniques for non-stationary data are discussed. Applying our framework to both real and synthetic traces we i) illustrate how to compare traces using trace-driven queueing, and ii) show that traces that look 'similar' under various statistical measures (such as the Hurst index) can exhibit rather different behavior under queueing simulation.

Key words: Trace-driven queueing, Queueing simulation, Long-range dependence, Visualization tools

1 Introduction

The remarkable discovery of self-similarity and long-range dependence properties in network traffic by Leland et al. [31] have stirred a large body of work over the last ten years. Numerous studies have established the failure of Poisson modeling for Internet traffic at various levels of aggregation (from local area networks to backbones, see [39]), and a large number of alternative
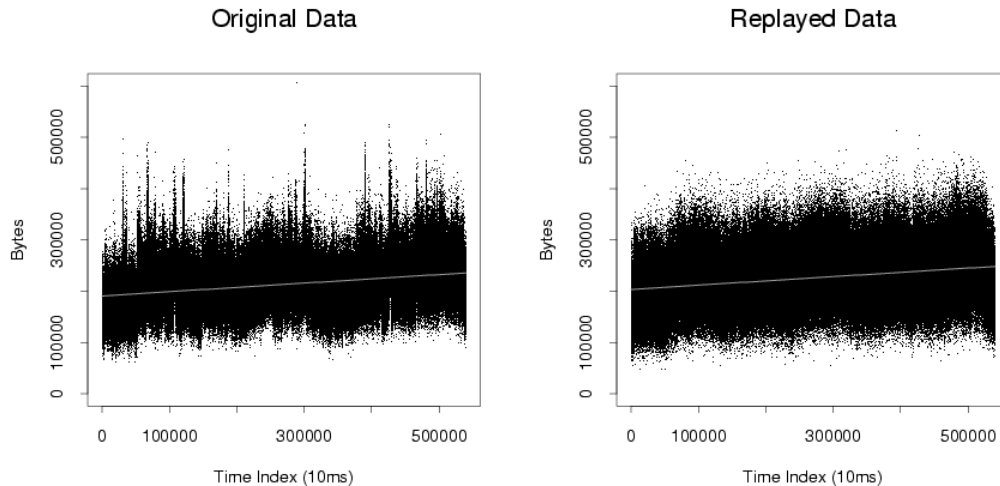
Original Data	Replayed Data

Fig. 1. The original data is less 'spiky' than the replayed data, but is it significant?

models have been proposed to capture the properties of packet and byte arrival processes (see [6,48] for overviews). However, the dependence structure of Internet traffic is such a complex phenomenon that no model is entirely satisfactory, and recent studies have even proposed a return to Poisson models for the smallest time scales in high-aggregation environments (see [5,26,52]). It is clear that much work and new modeling techniques are needed, and this fertile ground has created a growing interest by network researchers, applied probabilists and statisticians. Unfortunately, this puts researchers that are working on the development of new network mechanisms and algorithms in a difficult situation. In order to gain a good understanding of network traffic and obtain the needed mathematical and visualization tools, they are forced to navigate a very large and complex literature with few definitive truths.

In this paper, we argue that it is useful to go back to the well-established field of queueing theory to study network traffic. Our motivation is that queues provide the most intuitive language for describing traffic and its dependence structure, and furthermore, queue length, delay and loss are more meaningful performance metrics in the network context.

In this vein, during the evaluation of a new traffic generation approach developed at the University of North Carolina, we had to convince ourselves that the properties of the generated traffic were *close enough* to those of real traffic. The foundation of this approach is a method for transforming a packet header trace into an application-neutral, source-level representation that enables closed-loop traffic generation of arbitrary traffic mixes (see Section 4.1 below and [18] for more details). Figure 1 shows the sequence of binned byte counts from a real network trace from the Abilene backbone (labeled "Original Data") and a synthetic one (labeled "Replayed Data") generated in a network testbed using an early version of the traffic generation tool. While the two arrival processes are not visually identical, are they close enough that

the data on the right can be used for research purposes?

In order to further motivate our approach, we now examine one concrete domain, validation of synthetic network traffic, in which careful understanding and accurate modeling of network traffic is crucial. The networking community relies heavily on experiments run on simulators and testbeds in order to assess the performance of new network protocols, and synthetic traffic represents a key component of such experiments (see [14,51]). It is therefore very important to verify that the characteristics of synthetic traffic traces used in network experiments match well those of real traffic. Although providing a conclusive list of desirable properties is a difficult task, it is clear that reproducing the nature of the queueing process observed on real Internet links is fundamental for many research studies. For example, congestion control research (*e.g.,* [13,21,23,29]) focuses on developing new mechanisms that mitigate delay and loss in the face of network over-utilization. Since both delay and loss are functions of queueing dynamics, synthetic network traffic must result in a realistic queueing process in order to obtain relevant results from experiments. Realistic queueing is also needed in experiments that study, for example, router capacity planning (*e.g.,* [15]), quality-of-service guarantees (*e.g.,* [24]), and denial-of-service attacks (*e.g.,* [17]).

Figure 2 shows that the marginal distributions of the two traces are quite similar. Their means are also close: 158.6 Mbps for the original trace and 170.9 Mbps for the replayed one. Their standard deviations are not too different either: 37.4 Mbps for the original trace and 40.4 Mbps for the replayed one. We can conclude that the two marginal distributions are close enough for most purposes.

Figure 3 compares the scaling properties of the two traces using the *log-scale diagram* derived from wavelet analysis (see the work of Abry and Veitch [1,50] for details). Estimates for the Hurst parameters, H, and confidence intervals of the two traces are also given in the plots. One could easily argue that the
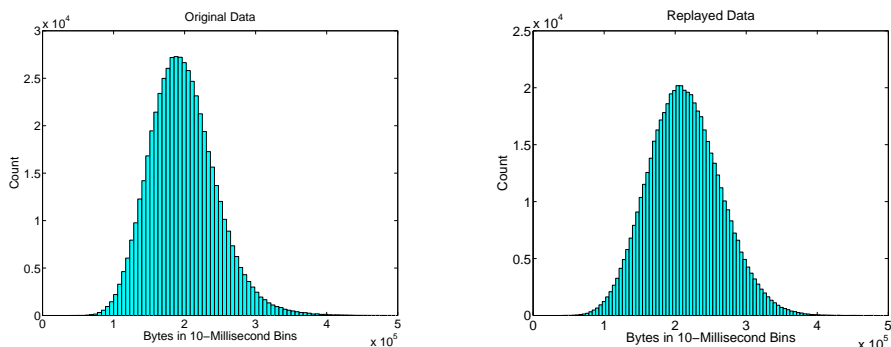

Fig. 2. The marginal distributions of the original and the replayed data are comparable.
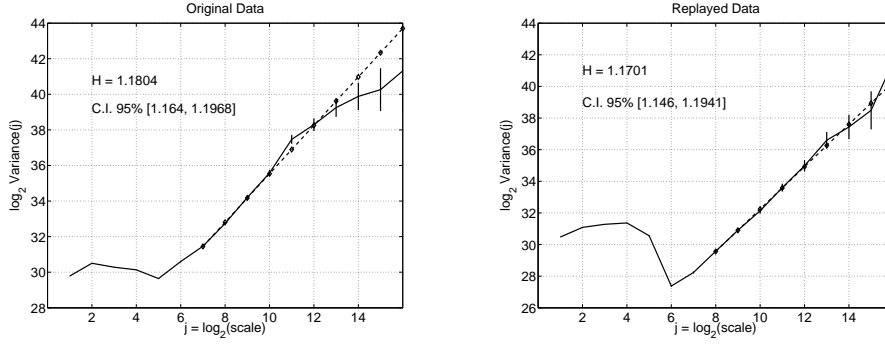
Fig. 3. The log-scale diagrams of the original and the replayed data reveal similar scaling behaviors, especially for the middle time scales.

two traces have similar scaling, especially for the middle time scales.

Did the previous rough comparison demonstrate that original and replayed traces have similar characteristics? Queueing analysis can clarify this question in a natural way. Results from trace-driven queuing simulations in Figure 4 (explained more fully in Sections 3 and 4) show that the nature of the two traces is completely different, since the original trace creates much longer queues (the shapes are completely different!). The queueing analysis makes it clear, in a much more obvious manner than other methods, that something was missing in this early attempt to generate realistic synthetic traffic.

In this paper we argue that an analysis using trace-driven queueing simulation is a useful tool for better understanding traffic characteristics. Our goal with this paper is to introduce a methodology for analyzing traces and the necessary tools and visualizations to do the analysis. We also show using real traces
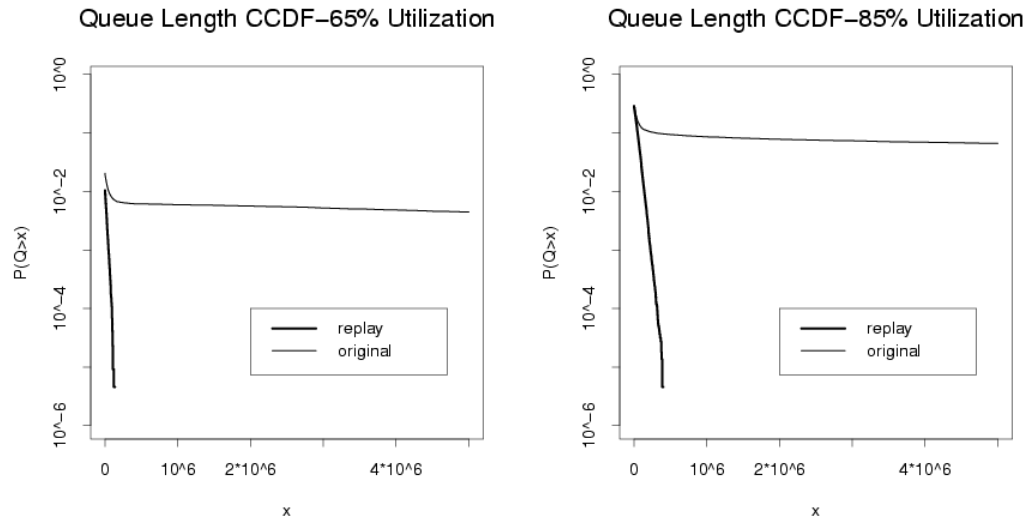

Fig. 4. The tails of the queue length CDFs are very different. The tail of the CCDF from the replayed data (thick line) decays very quickly compared with the original data (thin line). For the original data, longer queues are much more likely.

4

that the conclusions reached from this type of analysis may point to a very different behavior than the one elicited from a statistical analysis. Finally, a queueing analysis and the associated metrics (such as backlog and delay distributions, loss rates) is capable, to a large extent, of uncovering similarities and differences between traces, a useful result for network simulation and emulation studies.

However, trace-driven queueing simulation requires that the data analyst make several choices that can seriously impact the results. Such choices include the utilization rate in the infinite buffer case and its constancy over time, and in addition, the buffer size in the finite buffer case. In this paper we describe these choices and show through examples how they affect the conclusions reached. We also examine several queueing metrics and finally introduce novel visualization tools (such as the "multiscale excursion plot") that we believe can help the analyst in their investigations of network traces. So, while the use of trace-driven queueing simulation is not new (*e.g.*, [11] demonstrated the role of dependence on queueing), our theoretical overview, methodology, visualizations and caveats provide a more complete and systematic presentation than previously seen.

Despite our advocacy of trace-driven queueing simulation, we acknowledge one must not take its conclusions too far. Networks like the Internet which use TCP/IP feedback congestion [32,23,2] and flow control [42] are *closed loop* systems while trace-driven queueing is inherently *open loop*, using no feedback. The ability of TCP/IP sources to respond and adapt to available network resources is important [25], and not captured with trace-driven queueing. In fact, it is one of the motivations behind the UNC traffic generation experiments. But here we make the subtle distinction that our goal is not performance evaluation *per se*, but simply to make queueing-specific inferences about trace datasets that should create similar router workloads. So, while we vary the service rate of a simulated server to cause additional queueuing, we do not, for example, make claims about the engineering problem of how to size real link capacities to meet Quality of Service requirements in the real network from which the traffic arose. Our goal here is to provide a systematic methodology to study the traffic, not the system.

The rest of this paper is organized as follows. In the remainder of Section 1 the traces used in this paper are described. Section 2 provides an overview of queueing theory and discusses several canonical examples. Section 3 describes trace-driven queueing simulation, and provides a number of visualizations for both simulated and real network traces. Section 4 gives a more detailed queueing analysis of the original Abilene and replayed synthetic traces described above. Techniques for addressing certain non-stationary features are discussed. Section 5 describes problems and issues one can face doing trace-driven queueing analysis. In Section 6 we provide Conclusions.

1.1 Description of data sets used in the study and a motivating example

The data sets examined in this paper come from two sources: (i) the UNC data collection experiments and (ii) the Abilene network collection. Queueing analysis of the Abilene data set is our primary emphasis because of its role in refining a new UNC tool for capturing and replaying network traces. However, for variety and to illustrate particular ideas, we also discuss certain UNC data sets.

The 2002 UNC data sets are a collection of packet header traces obtained in April 2002 from the 1 Gbps Ethernet link connecting the campus of the University of North Carolina at Chapel Hill with the rest of the Internet. Only packets observed in the inbound (from the Internet) fiber link were considered. These traces were collected using the tcpdump tool running on a high-end Intel-based server-class machine (with loss rates of at most 0.1%). In order to perform the analyses reported in this paper, we transformed these traces into time-series of packet and byte arrivals for 10-millisecond intervals; this interval turns out to be well within the precision of our measurement infrastructure.

The second source of data are network traces from the Abilene-I collection of packet header traces that is publicly available from the National Laboratory for Advanced Network Research (NLANR) [35]. These traces were collected from Internet 2 backbone links (OC-48) in August 2002 using a DAG card [10]. The measured links carried traffic between Indianapolis and Cleveland, and between Indianapolis and Kansas City. As before, these traces were transformed into time-series of arrivals for 10-millisecond bins (this is certainly within the precision of the high-time-resolution DAG card).

A time-series plot of the packet counts of two Abilene traces is shown in Figure 5. One of the main features of both traces is the burstiness exhibited over different time scales. This burstiness is usually captured by the Hurst parameter, but as our subsequent analysis shows, its effect on queueing is more subtle.

2 A Brief Overview of Queueing Results

In this section we provide some background on queueing results, with a particular emphasis on models proposed to capture the characteristics of Internet traffic, such as Fractional Brownian Motion.

Consider time being divided into "slots" of fixed length (e.g., 10 ms). Denote the amount of work that arrives to a server (bytes or packets, for example) in
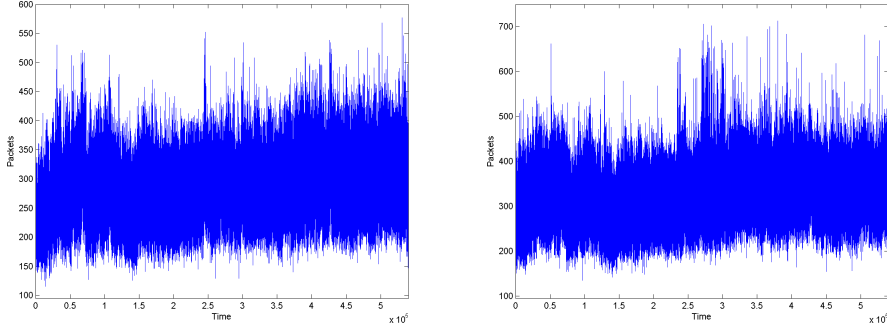
Fig. 5. Abilene packet traces: Indianapolis-Cleveland (left panel) and Indianapolis-Kansas City (right panel).

slot k by $X_k = Y_k + m$ where $E[Y_k] = 0$ and $m > 0$. Furthermore, it is assumed that Y_k is a stationary, ergodic process. In this formulation, the quantity Y_k captures the variability and m the mean amount of work that arrives in each time slot. Then the total cumulative input to the server in time $\{1, 2, \ldots, n\}$ is

$$A_n = \sum_{k=1}^{n} X_k.$$

Suppose that the server can process C units of work in each time slot. Then the net input in slot k is $X_k - C$ and a net input process can be defined as

$$I_n = A_n - Cn = \sum_{k=1}^{n} X_k - Cn.$$

The queue length process can be defined through the Lindley recurrence formula given by

$$Q_0 = 0 \text{ and } Q_n = \max\{0, Q_{n-1} + X_n - C\}, \, n = 1, 2, \ldots. \quad (1)$$

For $C > m$ (i.e., the average rate of the output/departure process exceeds the average rate of the input process) the queue length process has a proper steady-state distribution Q and the queue is characterized as *stable*. Moreover Q can be written in terms of the input process through

$$Q \stackrel{D}{=} \max_{0 \le k < \infty} \{A_n - Cn\} \quad (2)$$

where $\stackrel{D}{=}$ denotes equality in distribution. A queue with finite size B can be expressed as

$$Q_0^{(B)} = 0 \text{ and } Q_n^{(B)} = \min\left\{\max\{0, Q_{n-1}^{(B)} + X_n - C\}, B\right\}, \, n = 1, 2, \ldots. \quad (3)$$

(The books by Feller [12] and Asmussen [3] provide a theoretical background of queueing, with the case of i.i.d. input sequences receiving very detailed

7

coverage. For an exposition of results under a more general stochastic setting see the book by Bremaud and Baccelli [4].)

This discrete time framework would be the basis for processing real network traffic traces and assessing their properties. However, some of the theoretical results come from an analogous formulation in continuous time. Let the total input to the server in $[0, t)$ be $A(t)$, a stochastic process with stationary increments. Suppose that the server can process work at rate C. Then the net input process for work arriving in $[0, t)$ is

$$I(t) = A(t) - Ct.$$

The continuous time analog of the queue length process is the "workload" process, also called the "storage process" [16,36,37]

$$W(0) = 0 \text{ and } W(t) = \sup_{s \leq t}\{I(t) - I(s)\}. \tag{4}$$

In order for the queue to be stable, we need that $E[A(t) - Ct] < 0$. Then the distribution of the steady-state workload can be expressed as

$$Q \overset{D}{=} \sup_{t \geq 0}\{A(t) - Ct\}. \tag{5}$$

Note the similarity between (2) and (5). The three examples that follow provide some queueing results for several canonical families of traffic models. They assume finite variance distributions, although infinite variance models have also been proposed (*e.g.*, [27]).

2.1 Example - Brownian Motion

Let $\{B(t)\}$ be standard Brownian Motion (i.e., $\text{Var}[B(1)] = 1$) and

$$A(t) = \sigma B(t) + mt, \ m > 0.$$

Assume for the server rate C that $m < C$. Then the net input to the server is

$$I(t) = \sigma B(t) + (m - C)t,$$

a Brownian Motion with negative drift. The steady-state workload has a distribution given by

$$Q \overset{D}{=} \sup_{t \geq 0}\{\sigma B(t) + (m - C)t\}$$

which is the supremum of a Brownian Motion with negative drift. Its distribution is known [28]. In fact,

$$\mathrm{P}(Q < x) \overset{D}{=} \mathrm{P}(\sup_{0 \leq t < \infty} I(t) \geq x) = \mathrm{e}^{-\lambda x}, \ x \geq 0$$

where $\lambda = \frac{2(C-m)}{\sigma^2}$. So in this case the supremum has exactly an exponential distribution. This is illustrated below in Figure 9 of Section 3 where several useful visualizations are described.

2.2 Example - Weak Dependence with Finite Moments

In Glynn and Whitt [16], Lindley processes (i.e., processes defined by (1) or (4)) were considered using classical large-deviation techniques (see also [9,30]). Suppose $A(t)$ is a stochastic process with stationary increments. (A discrete time analog was also considered.). Moreover assume $E[I(t)] = at$, $a < 0$. Under some conditions that require the dependence of $I(t)$ be not too strong ('weakly-dependent') and that $I(t)$ have finite moments, the distribution of the steady-state queue length Q obeys

$$\lim_{x \to \infty} \frac{1}{x} \log \mathrm{P}(Q > x) = -\theta^* \text{ for a constant } \theta^*.$$

The conditions on the dependence and moments are violated if the input process has long-range dependence or infinite variance, respectively. Notice that this result on the log-asymptotics is weaker than saying the workload has an exponential distribution.

2.3 Example - Fractional Brownian Motion

Fractional Brownian Motion (FBM) has been proposed as a Gaussian traffic model [36,37] for the variability in the cumulative arrivals that allows long-range dependence in the increments (e.g., the binned packet or byte counts). Indeed the model has received much attention (see Norros [38] for a list of references), in part because of its simplicity. Including the mean and variance, the simplest FBM model would have only three parameters. (See the papers in Doukhan et. al. [7] for theory and applications of long-range dependence.)

Let $\{B_H(t)\}$ be standard Fractional Brownian Motion with Hurst parameter $H \epsilon (0.5, 1)$. That is, a mean zero, self-similar Gaussian process $\{B_H(t)\}$ with stationary increments and covariance function

$$\mathrm{Cov}[B(s), B(t)] = (|t|^{2H} + |s|^{2H} - |t-s|^{2H})/2.$$

In the time interval $[0, t)$ denote the cumulative input by $A(t)$ and the net input process by $I(t)$ where

$$A(t) = \sigma B_H(t) + mt \text{ and } I(t) = \sigma B_H(t) + (m - C)t$$

9

so that Q is given by (5). For modeling packet or byte counts in, say, 10 ms intervals, one would imagine using the increments $A(t + 1) - A(t) = \sigma(B_H(t + 1) - B_H(t)) + m$. The increments of Fractional Brownian Motion $\{B_H(t + 1) - B_H(t)\}$ are called Fractional Gaussian Noise (FGN), and are long-range dependent for $H \in (0.5, 1)$. In Duffield and O'Connell [8] classical large deviation results were extended to show that for this model

$$\lim_{x \to \infty} \frac{1}{x^{2-2H}} \log \mathrm{P}(Q > x) = -c_2, \ c_2 > 0 \text{ a known constant.} \tag{6}$$

Now, for a generic random variable W with a Weibull distribution, shape parameter $\alpha > 0$, and scale parameter $c > 0$

$$P(W > x) = e^{-cx^\alpha}, \ x \geq 0 \text{ and } \lim_{x \to \infty} \frac{1}{x^\alpha} \log P(W > x) = -c. \tag{7}$$

Because of the similarity of the limits in (6) and (7) it sometimes said that the workload from Fractional Brownian Motion input is "Weibull-like". However, this log-asymptotic result does not say the steady-state workload has a Weibull distribution. The difference is the lower order terms that are washed out in the limit. (See [22,33] for more details.) The important consequence for queueing is that the tail of the queue length distribution decays more slowly than an exponential rate, so larger buffers are more likely than from a similarly scaled weakly-dependent input process.

3   Trace-Driven Queueing Analysis

In order to perform trace-driven queueing, the arrival (or workload) sequence $\{X(k); k = 1, 2, \cdots, N\}$ of work per unit of time (e.g. the number of bytes or packets per 10 ms intervals) can be used. The arrival/workload sequence can be either a real or simulated traffic trace. The user defined parameter C determines the utilization rate, ρ, of the simulated server through the formula $\rho = E[X(n)]/C$. To force queueing, it is not uncommon to use values of ρ at 90% or higher as a means to understand the burstiness of the trace. We do not claim that the performance at a simulated 90% utilization is comparable to a real queue with 90% traffic intensity.

The relationship between the utilization rate and the mean rate is clearly illustrated in the following example. The left panel of Figure 6 shows a simulated Fractional Gaussian Noise trace of length $n = 7,200,000$, with Hurst parameter $H = 0.9$, mean 30 and variance 25. The horizontal gray lines show (from bottom to top) the sample mean (100% utilization), the 90% utilization line and the 65% utilization line. With an empty queue, only an amount of work above the utilization line would produce an onset of queueing at that

10

utilization. In the right panel of Figure 6 a magnified version of the previous figure showing only 10,000 data points is given. Notice how infrequently the arriving work exceeds the 65% utilization line.
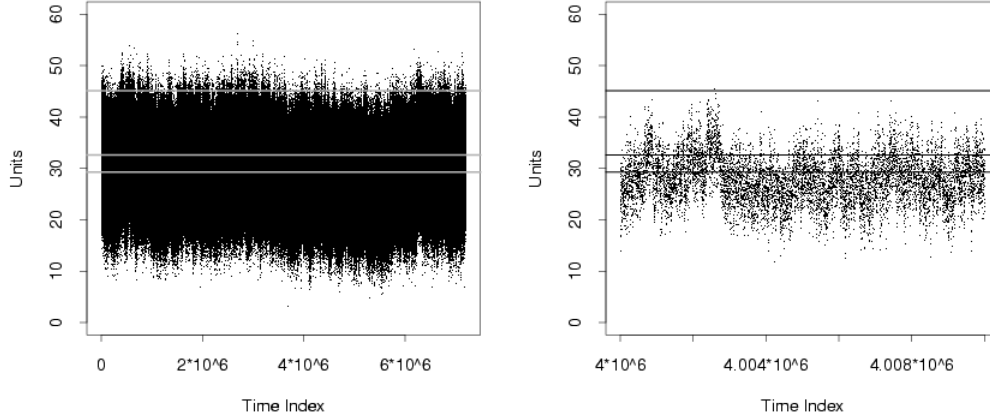

Fig. 6. Simulated Fractional Gaussian Noise with Hurst parameter $H = 0.9$, mean 30, and variance 5. The entire dataset (left) has length 7,200,000 while the subset starting at 4×10^6 (right) has only 10,000 points. Horizontal lines correspond to (bottom to top) the sample mean, the server rate for 90% utilization, and the server rate for 65% utilization.

When comparing two traces, even small differences in the mean rates could cause quite different queueing behavior if the same server rate C is used. The results of the anaylsis could be dominated by the difference in the mean rates. We propose fixing the utilization rate ρ when comparing traces. Loosely speaking, this has the interpretation of a server allocating resources in proportion to the mean rate. We adopt this method throughout this paper.

We next introduce some queueing performance metrics and describe the associated plots that help with the interpretation of the results.

3.1   Queue Lengths and the Corresponding Time Plot

Assuming no restriction is placed on the maximum size of the queue, the most fundamental output quantity is the sequence of queue lengths $\{Q(n); n = 1, 2, \cdots, N\}$, obtained through Lindley's recursion equation (1). A useful plot that provides a global view of the effect of the utilization rate is the queue length time plot, that plots $\{Q(n)\}$ against time indices n. Figure 7 shows the queue lengths corresponding to a 65% utilization (left panel) and 90% utilization (right panel). While the increase in the utilization is only about 38%, the queue lengths have increased by many orders of magnitude. In fact, at 65% utilization there is little queueing at all. At 50% utilization there is

11

practically no queueing with this trace (not shown). Finally, notice how periods of higher activity in the input process around time index 1×10^6 and 3×10^6 (Figure 6 - left panel) and lower activity around 5×10^6 translate into larger and smaller queue lengths, respectively, at the same times.
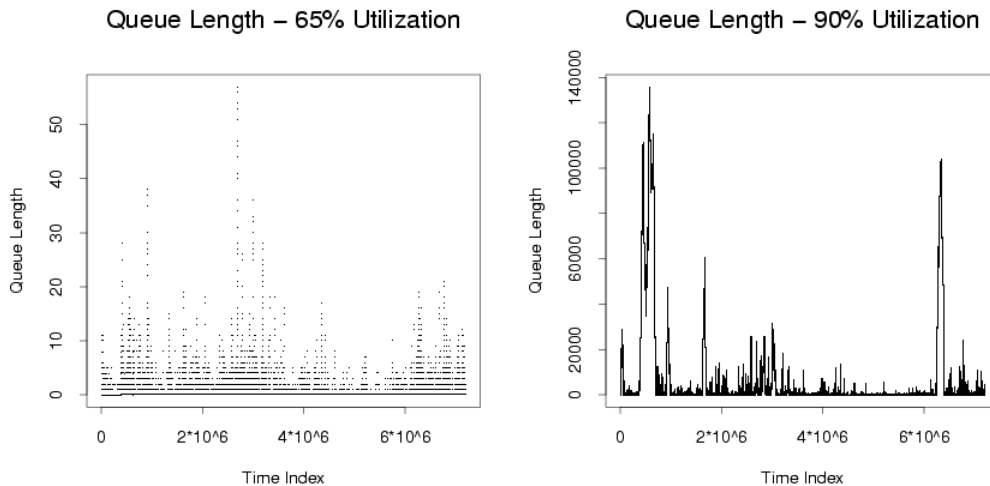

Fig. 7. Queue lengths from a simulated queue with simulated Fractional Brownian Motion ($H = 0.9$) input and 65% utilization (left) and 90% utilization (right).

A combination of time plots, all aligned together offers the data analyst a more complete queueing picture. Figure 8 shows three related plots for a queueing simulation using the Abilene Cleveland dataset. The top panel shows the input process $\{X(k)\}$. (For this queueing simulation the server rate C was calculated using not the overall sample mean, but the 1 s "local means", calculated as the sample mean on non-overlapping 1 s intervals. See our discussion in Section 4.) In the middle panel, the evolution of the queueing process $\{Q(k)\}$ over time is shown. In the bottom panel, the departure process is plotted. The departure process is the amount of work that leaves the queue in each time interval. From the combined plot, it can be seen which bursts in the input process are translated to "spikes" in the queue length process and to corresponding bursts in the departure process. The figure also allows users to understand how queueing shapes an input process and which features are preserved, dampened or enhanced in the departure (output) process.

3.2 Queue Length CCDF Plots

From the sequence of queue lengths, statistics such as average queue length or queue length distribution can be obtained. In order to obtain an accurate estimate of these statistics a certain percentage of the initial queue lengths (burn-in period) should be excluded. For stationary input processes, experience shows that a 1-10% proportion (depending on the length of the trace)
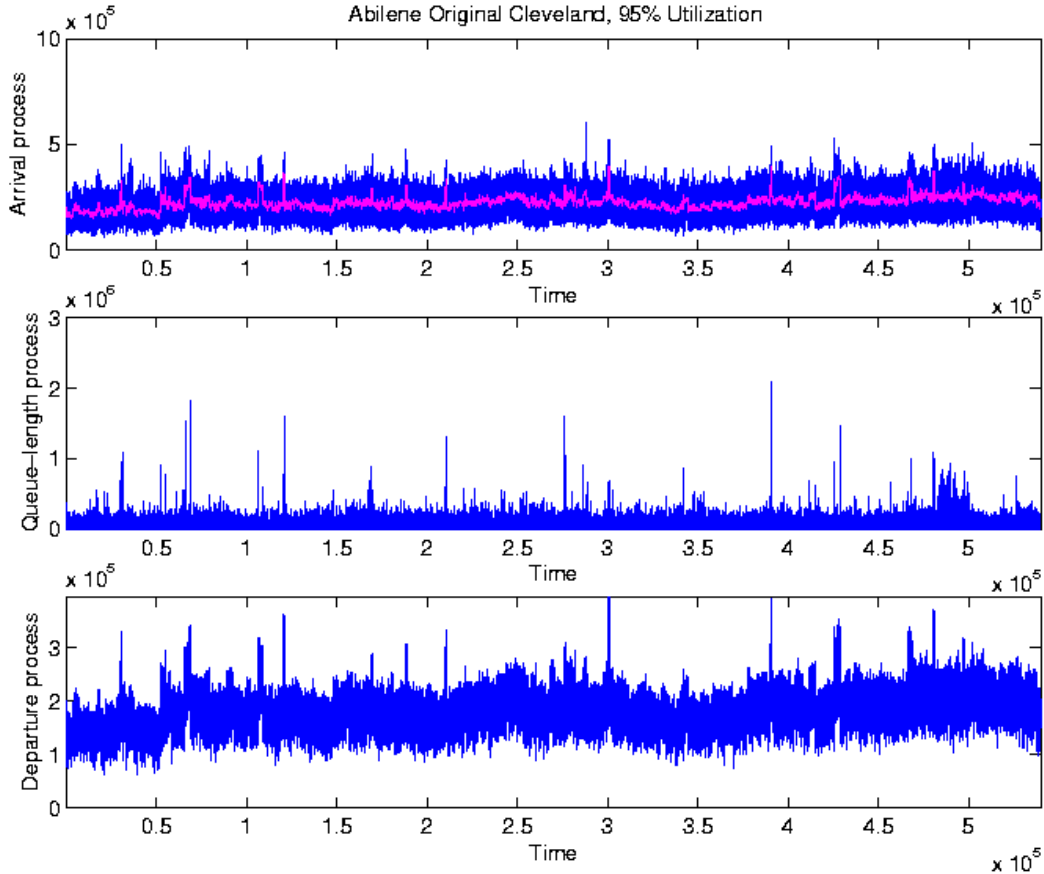
12

Fig. 8. Indianapolis-Cleveland packet trace (top) and associated queue length (middle) and departure (bottom) processes using a server rate adapted to 1 s local means. Features in the input process can be clearly connected to corresponding features in the queue-length process.

is adequate for the task. Two techniques to deal with certain non-stationary input sequences are described in Section 4.

Figure 9 (left panel) shows the tail of the empirical queue length CDF (the CCDF) using two Gaussian input sequences, both with mean 30, variance 25 and length $n = 7,200,000$. One is simply an independent and identically distributed (iid) Gaussian sequence, while the other is simulated Fractional Gaussian Noise with Hurst parameter $H = 0.9$ generated using a variation on Paxson's fast approximate technique [40,45]. Each of the values in this simulated trace were truncated to give integers, and then fed into a queue with 65% utilization rate. The distribution was obtained after excluding the first $50,000$ queue lengths. As is common, the vertical axis is shown in log-scale to emphasize the rate of decay. A straight-line in this plot would correspond to exponential decay in the tail of the queue length distribution, which we see for the i.i.d. sequence. On the other hand, the long-range dependent FGN input sequence provides a distinctive curve which means larger buffers are more likely. Looking at the x-axis reveals that the probability the queue length

13

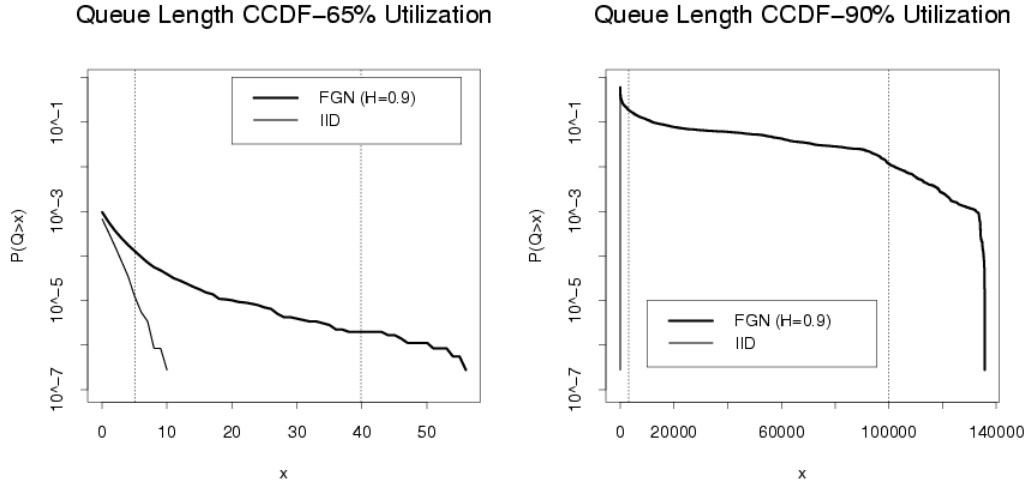exceeds size 10 is less than 10^{-6} for the i.i.d. sequence but around 10^{-4} for the FGN sequence.

Fig. 9. Empirical queue length CCDF for two simulated N(30,25) Gaussian sequences: i.i.d. (thin line) and Fractional Gaussian Noise ($H = 0.9$), (thick line) fed into a queue with 65% utilization (left) and 90% utilization (right). Long-range dependence means larger queues are more likely.

In Figure 9 (right panel) an analogous plot is shown for the same i.i.d. and Fractional Gaussian Noise data processed by a server operating at 90% utilization. Once again, the resulting graph for FGN is not a straight line and shows a typical "rotated-S" shape. The curvature for small queue lengths is consistent with long-range dependence, and is an important feature of the trace revealed by the CCDF plot. The drop-off at the extreme right of the graph is an artifact of the finite-length of the trace and is a typical characteristic. The queue lengths for i.i.d. data are so small in comparison that the queue length CCDF appears to be a vertical line. Comparing the queue length CCDF of the FGN data for the two utilizations, queue lengths for 90% utilization are longer with higher probability, as expected. For example, the probability that the queue length is greater than 20 is about 10^{-5} at 65% utilization, but more than 10^{-1} at 90% utilization.

When the cumulative input process $A(n)$ is Fractional Brownian Motion, an asymptotic relationship between the curvature of the plot and H is given in (6). If one makes the approximation

$$\log P(Q > x) = -c_2 x^{2-2H}$$

then

$$\log_{10}(-\log(P(Q > x))) = \log_{10} c_2 + (2 - 2H) \log_{10} x. \qquad (8)$$

So in practice, one would hope that for an approximately Fractional Gaussian

14

Noise input process, a graph of $\log_{10}(-\log(P(Q > x)))$ vs $\log_{10} x$ would show, for large enough queue lengths a linear region with slope $2 - 2H$. Figure 10 (left panel) shows such a plot using the same empirical CDF shown above for 65% utilization. A least-squares line fit for $0.7 \leq \log_{10} x \leq 1.6$ yields a slope of 0.179 which gives a Hurst parameter estimate $\hat{H} = 0.91$. Vertical lines corresponding to $log_{10}x = 0.7$ and $\log_{10} x = 1.6$ are included in Figure 9 (left panel) to provide a sense of scale. Notice that most of the region is included.

Fig. 10. Graph of $\log_{10}(-\log P[Q > x])$ vs $\log_{10} x$ to exploit the approximation described in (8). The slope of the linear portion can be related to the Hurst parameter.

Figure 10 (right panel) tries to exploit the same approximation, but for a 90% utilization rate. It is harder to fit a straight line on this plot. A least-squares regression line for $3.5 \leq \log_{10} x \leq 5$ gives an estimated Hurst parameter of $\hat{H} = 0.87$. This interval is shown in Figure 9 (right) by the vertical lines. Knowing that the graph should have a Weibull-like shape, Figure 9 (right) appears unreliable for x larger than about 90,000. The regression in Figure 10 (right) might be slightly improved by using $\log_{10} 90,000$ for the right endpoint instead of 5. On the other hand, the visualization in Figure 10 provides a way to see where the Weibull approximation is appropriate in the tail simply by looking for a straight line.

3.3   QQ-plots

Another useful plot, especially when comparing two different data sets (e.g. two queue-length processes from different times) is the QQ-plot. Figure 11 shows a QQ-plot of the queue length process for the original Abilene Indianapolis-Cleveland traffic trace and a related synthetic version produced by the UNC group. (See Section 4 for more details.) In this simulation, a 95% utilization rate was used. If the data were approximately a straight line it would

15

suggest the distribution of queue lengths from the two traces are similar. Instead, significant differences in the right tail of the queue length distributions are very clear. This indicates the synthetic traffic typically exhibits less burstiness, which results in smaller queues and a smoother queue length process.
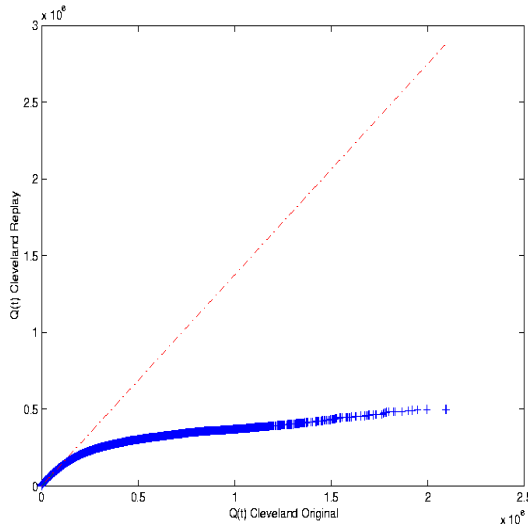

Fig. 11. QQ-plot of original and synthetic traffic traces for the Indianapolis-Cleveland link shown with 45 degree line. Since the curve is far from the line, the queue length distributions are clearly quite different.

3.4 Multiscale Plots

All the plots presented so far provide a 'snapshot' view of the effects of queueing on traffic traces at a single utilization. It is, however, particularly useful to obtain a more integrated picture of the whole process and the objective of the following plots is to capture such effects.

Figure 12 shows the mean queue length versus utilization from trace-driven queueing simulations using the original Abilene Cleveland trace and the UNC replayed version described in the Introduction. (See Section 4 for details on how the simulation was performed using a linearly-increasing server rate fit to the mean trend.) The growth of the mean queue length with utilization is expected. As utilization increases, server rate decreases, and queues become longer. But this plot also shows the the average queue length is larger using the original trace at larger utilizations. Although the vertical scale only highlights the differences at large utilizations here, smaller differences exist at utilizations between 50% and 80%. So this plot reveals, in absolute terms, the differences in the queue length, particularly at 90% utilization and above.
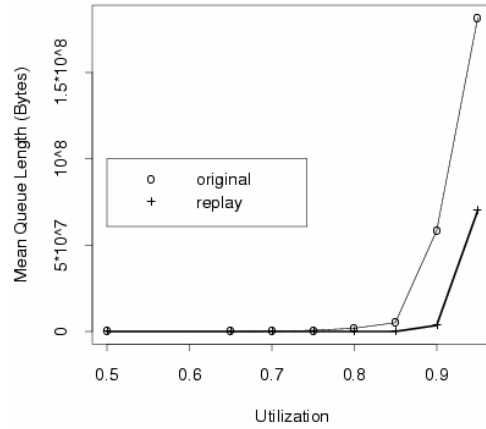
16

Fig. 12. Average queue length versus utilization for the original Abilene Cleveland trace and the UNC replayed version. Queueing differences at several utilizations are shown together.

Figure 13 shows similar plots for another comparison- the Abilene Indianapolis-Kansas City trace and its UNC replayed version. These plots use not the average queue length, but rather the empirical 95-th percentile (left panel) or 99-th percentile (right panel) of the queue length distributions, to provide analogous information. From the left plot we see that the queue length is zero at least 95% of the time when the utilization is 70% or less. The right plot shows that the queue length is zero at least 99% of the time when the utilization is 65% or less. As with the mean queue length plot, we can see for which utilizations the difference in the queue length distibution becomes large.
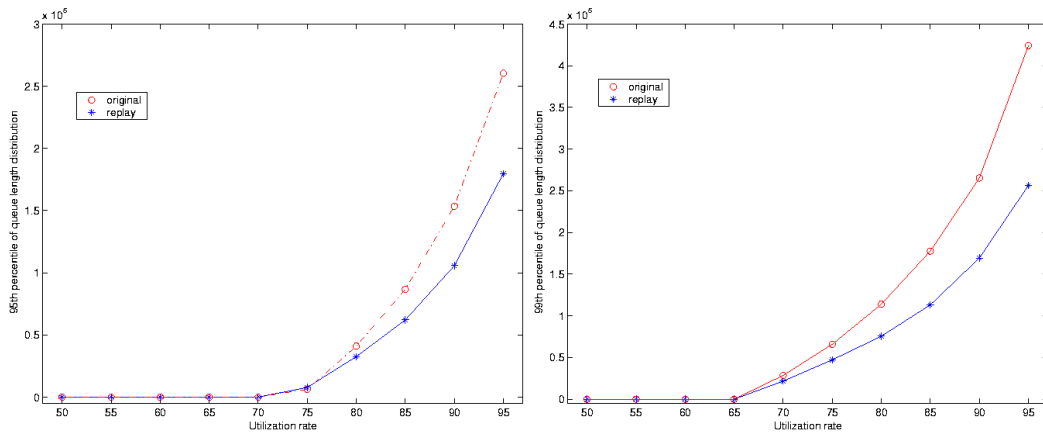

Fig. 13. 95th (left panel) and 99th (right panel) percentiles of the queue length distribution vs utilization rate for the original Abilene Indianapolis - Kansas City trace and the corresponding UNC replayed version.

17

In this section we present a novel visualization, a multiscale excursion plot, that illustrates the effect of using different adaptive service rates. Let $E_\ell(t)$ denote the *excursion* process at level ℓ of the corresponding queue length process $Q_\ell(t)$ for $t \in [0, T]$. The level parameter ℓ determines the service rate *process* $C_\ell(t)$ used for processing through the trace through the queue. For example, when $\ell = 1$ the service rate process $C_1(t) = $ constant and is proportional to the mean of the entire arrival process. This service rate $C_1(t)$ gives rise to a corresponding queue length process $Q_1(t)$ and corresponds to the usual definition of a constant server rate C. When $\ell = 2$ the trace is split into two parts on intervals $[0, T/2]$ and $(T/2, T]$ of equal length and the service rate process $C_2(t)$ takes two different values: $C_2(t) = c_1$ if $t \in [0, T/2]$ and $C_2(t) = c_2$ if $t \in (T/2, T]$, with c_1 and c_2 being proportional to the mean in each of the two halves of the arrival process. This in turn defines $Q_2(t)$. Continuing to further subdivide the trace, we get that when $\ell = 10$, the trace is divided into 2^{10} equal parts and the service rate process is given by the step function $C_{10}(t) = c_j, \ j = 1, \cdots, 1024$ if $t \in ((j-1)T/2^\ell, jT/2^\ell]$. Hence, at higher levels the service rate process becomes more *adapted* to *local fluctuations* of the arrival process through its proportionality to the sample mean on the sub-intervals.

The excursion process $E_\ell(t)$ identifies values of the queue length process with one of $K + 1$ states (colors) based on the magnitude of $Q_\ell(t)$. More precisely, $E_\ell(t)$ takes values in a finite set $\mathcal{K} = \{0, \cdots, k, \cdots, K-1\}$ as follows: $E_\ell(t) = k$ if $Q_\ell(t) \in [\tau_k, \tau_{k+1})$, where $\tau_k, \ k \in \mathcal{K}$ are predefined thresholds, with $\tau_K = \infty$. For illustrative purposes $\mathcal{K} = \{0, 1, 2\}$ is used here. Then, the excursion process $E_\ell(t)$ is plotted for several values of the level parameter ℓ.
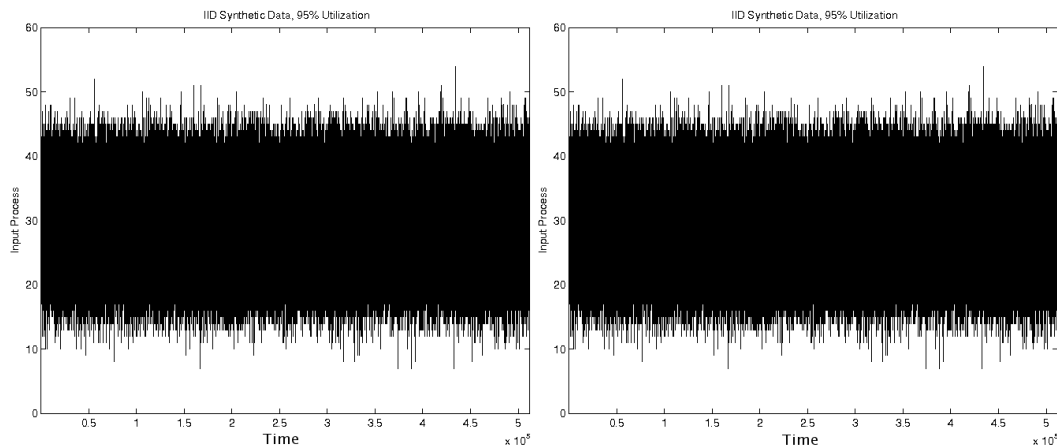

Fig. 14. Independent and identically distributed N(30,25) arrival process (left) and Fractional Gaussian noise N(30,25) (H=0.9) arrival process (right).

Figure 14 shows a subset from $n = 720,000$ to $n = 1,230,000$ of the same

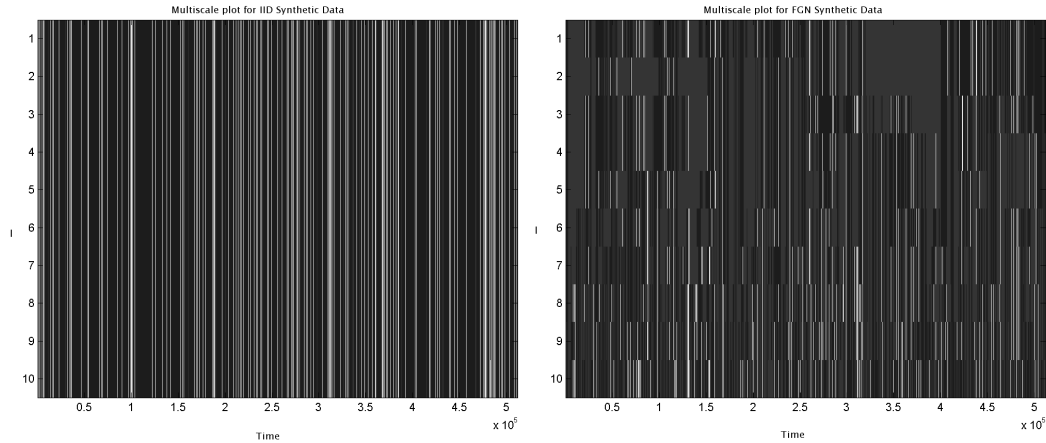Fig. 15. Multiscale plot of the excursion process $E_\ell(t)$ for the sequences shown in Figure 14 above.

N(30,25) simulated traces considered for Figure 9. Figure 14 (left panel) shows the i.i.d. sequence and Figure 14 (right panel) shows a sequence generated from a fractional Gaussian noise model with Hurst parameter $H = 0.9$. The multiscale plots for the excursion process corresponding to these simulated traces are shown in Figure 15. The threshold values used for both traces are 10 and 25; i.e. if $Q_\ell(t) \leq 10$ then $E_\ell(t) = 0$ (black), if $10 < Q_\ell(t) \leq 25$ then $E_\ell(t) = 1$ (white), otherwise $E_\ell(t) = 2$ (grey). It can be seen that for the i.i.d. trace the excursion process exhibits relatively few excursions above 10, but more importantly these excursions have a very short duration *at all* levels. This is due to the fact that the underlying input process has no memory for all practical purposes. At all levels there is a very short and not persistent burst that dies down immediately. The 'local means' are very similar to the 'grand mean' and hence the same pattern occurs at all levels. Any excursions are due to the fluctuations of the trace. On the other hand, the FGN trace exhibits long-lasting excursions above the thresholds for low levels, which is due to the temporal dependence present and the lack of adaptability of the rate process to local persistent fluctuations, as can be seen in Figure 14 (right panel). However, such excursions become less prominent and short-lived at large levels (above 6) due to the adaptability of the rate process. Nevertheless, their durations are still longer than those of the i.i.d. trace at the same level. Using the excursion process $E_\ell(t)$ obtained from trace-driven queueing for testing properties of the input process is a topic of ongoing research.

3.6 Finite Buffers and Loss Rates Plots

The presentation so far has assumed an infinite buffer. With a finite buffer, packets arriving to the server that cannot be accommodated by the existing free space in the buffer are dropped and lost. Therefore, values of the queue

19

length/workload process and the departure process are truncated. In this case, the loss process becomes of interest. The loss process (i.e., the sequence of counts of lost bytes or packets) is defined as $L(t) = \max\{0, Q(t) - B\}$, where B denotes the buffer size;. The magnitude of the loss process is obviously a function of both the utilization rate and the size of the buffer.

Figure 16 shows the loss process for the Abilene Indianapolis-Cleveland data time-aggregated to 1 second intervals. Figure 16 (left) shows the loss process for a buffer size $B = 20,000$ while Figure 16 (right) shows the loss process for a buffer size $B = 200,000$. Notice that the smaller buffer leads to larger and persistent loss rates (always above 1.5%). On the other hand, a 200Kb buffer to a large extent identifies large excursions of the input process above its mean level.
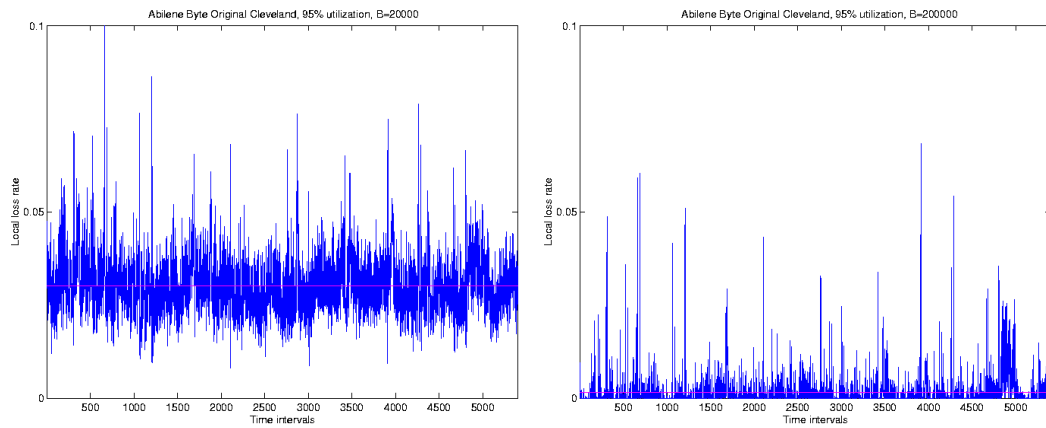

Fig. 16. Loss rate process for the Indianapolis-Cleveland traffic traces for a finite queue of size B=20000 (left panel) and B=200000 (right panel).

In the finite buffer case there are two factors involved in the queueing processing of network traces: the utilization rate ρ and the buffer size B. A 3-dimensional plot that captures the combined effect of both factors for the Abilene Indianapolis-Kansas City trace is shown in Figure 17. It can be seen that there exists a trade-off between buffer size and utilization rate; that is, as the utilization rate increases, a larger buffer size is needed in order to keep the average loss rate below a certain level. It is also worth noting that for utilization rates below 90%, a relatively small buffer size of about 20Kb manages to keep the average loss rate below 2%.

4 Case Study: Analysis of Original and Replayed Traffic Traces through Queueing

As mentioned in the Introduction, one of the motivations of this paper was to describe the use of queuing simulation for studying the realism of syn-
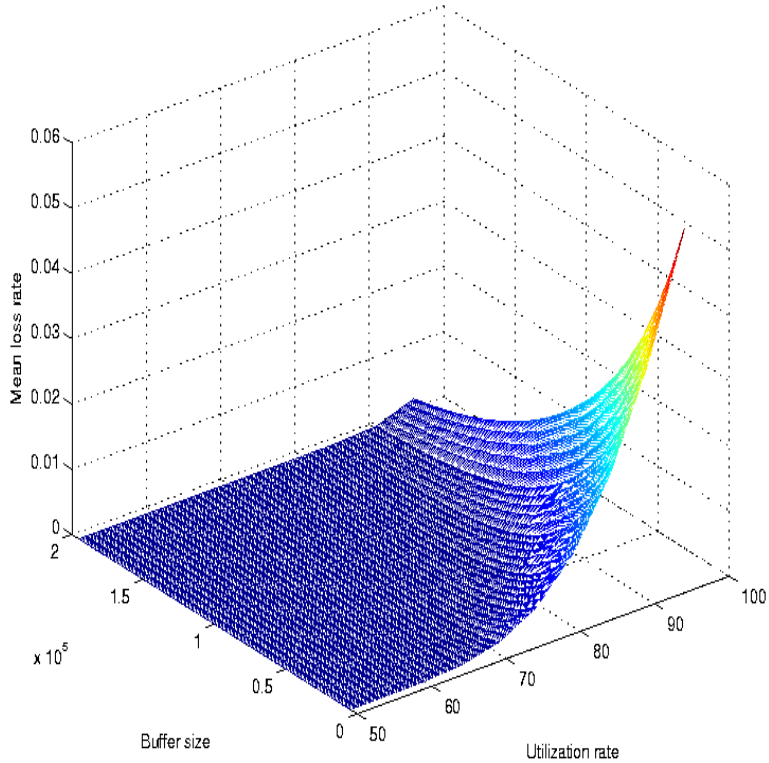
Fig. 17. Mean loss rate for the Indianapolis-Kansas trace at different utilization rates and for different buffer sizes. The tradeoff between buffer size and utilization rate is clearly shown.

thetic network traffic. In this section we describe ongoing network experiments performed by one of the authors (Hernandez-Campos) which attempt to reproduce the relevant statistical properties using a new traffic generation approach. Following a description of the experiments we show the results of two comparisons using trace-driven queueing analysis. The first comparison, using an earlier version of the traffic generation tool, revealed that some important property of the original traffic was not captured in the experiment, which provided motivation to refine the approach.

4.1 Description of the Network Experiment

The traffic generation experiments studied in this paper make use of the new *source-level trace replay* approach recently developed by Hernandez-Campos, et al. [20,18]. The starting point of this approach is the paradigm of simulation based on source-level descriptions of applications advocated by Floyd and Paxson [14]. These authors observe that TCP congestion control is an end-to-end *closed-loop* mechanism that shapes the low-level packet process according to the conditions of the network. Consequently, traffic must be generated by models of applications layered over (real or simulated) TCP/IP

21

protocol stacks. This is in contrast to an *open-loop* approach in which traffic is generated according to a model of packet arrivals, ignoring congestion control. Since between 90% and 95% of the bytes over the Internet are carried by TCP connections, simulations and testbed experiments must make use of synthetic traffic generated in a closed-loop fashion.

Source-level trace replay is based on a number of measurement techniques that can be used to convert an arbitrary packet header trace into a set of source-level descriptions of the connections in that trace. Any TCP connection, independent of the specific application driving it, is not more than a sequence of data exchanges, in which each end point sometimes sends data, sometimes receives data and sometimes remains quiet. Using fields in the header of each packet in a TCP connection, we can construct a description of this send/receive/wait behavior. TCP headers provide enough information to avoid being misled by packet retransmissions and reordering, so the description of traffic is in terms of *application-data units* that are independent of the network conditions in the trace. A trace can be *replayed* in a closed-loop fashion by first converting each connection to this abstract representation, and then using this same representation to drive an equal number of connections, maintaining the same relative connection start times. The major advantage of this approach is that it will enable re-creation of "realistic" traffic, in a reproducible way, in a closed loop fashion. By construction, this synthetic traffic is realistic at the source-level, since we directly measured and replayed the source-level properties of the entire population of TCP connections. However, does this mean that the traffic is "realistic" at the packet arrival level? One way of studying this question is to leverage on the nature of this traffic generation approach to directly compare the packet arrival processes in the original and in the replay trace. These two processes are obviously not identical, but are they "close enough"? Trace-driven queueing analysis can help answer this question in a network-centric manner (since queue lengths are very meaningful for network researchers and practitioners).

The replayed traces considered below were generated in the UNC testbed environment shown in Figure 18. This testbed consists of more than 40 traffic generation machines, and a number of switches and routers. During the experiments, instances of the `tmix` traffic generation program are used to replay the source-level behavior of the connections in the original trace. In the case of the replay of the 2-hour-long Abilene trace, which has 2.4 million TCP connections, each traffic generator was in charge of reproducing 60,000 connections. In other words, each traffic generator opened or accepted 60,000 TCP connections (preserving the original relative start times) and then sent/received/waited on each connection according to its original source-level behavior. We verified that CPU and other resources were not constraining the traffic generators. In order to introduce realistic delays in the experiment, we made use of the *dummynet* system [44] to assign random delays to each

connection (so all packets from the same connection were delayed by the same randomly-chosen amount of time).
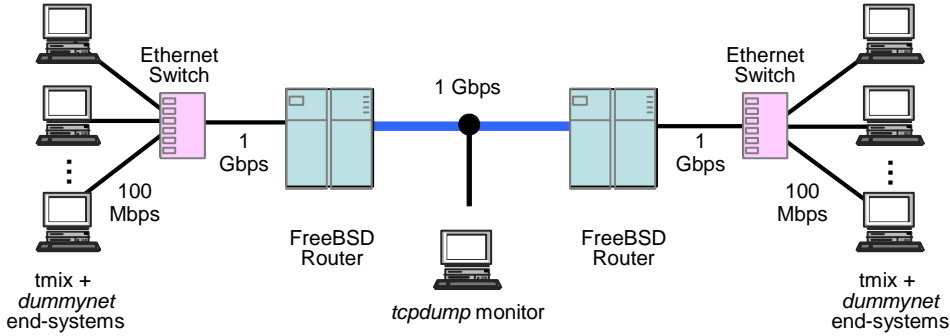

Fig. 18. Configuration of UNC Network Testbed

4.2   Comparison of Traces using Trace-Driven Queueing

In this section we provide in-depth analyses of the comparison between original and UNC replayed traces. In the first comparison, the Abilene trace was compared with its replayed version. As described in the introduction, the traces consist of 540,000 byte counts measured at 10 ms intervals. Figure 19 shows the original Abilene trace (left panel) and a replayed version (right panel) from the network experiment. A trend line is apparent, and is discussed below. The replayed trace contains certain features of the original data, but new features of randomness were introduced. A close examination of the two time plots reveals the traces are not identical (although that was the goal). The original data appear 'more bursty'. But, is the replay trace similar enough for network research purposes, such as investigations into active queue management protocols? Trace-driven queueing simulation can suggest an answer.

One of the issues that must be dealt with is the upward trend in the traces. If a queue with constant service rate were simulated over the length of the trace, the utilization measured locally (i.e. over short non-overlapping time intervals) would increase over time. This in turn would translate into larger queue lengths towards the end of the trace. In order to isolate the queueing effects of the 'spikiness' and dependence structure in the data, we suggest a variable server rate $C(k)$ and discuss two approaches.

The first approach is only appropriate for traces with a clear linear trend and a variance that does not increase too much as the mean rate increases. Using linear regression, a line of the form $y = mk + b$, with slope m and y-axis intercept b is fit through the data. Values for m and b are shown in Figure 19 for the two traces. We obtain the adapting server rate using $C(k) = (mk+b)/\rho,$
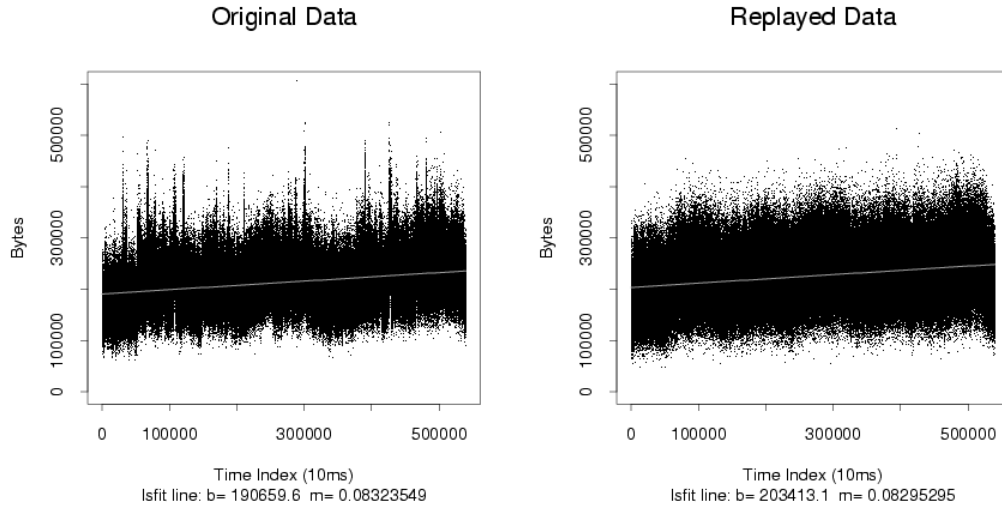
Fig. 19. Byte counts of original and replayed data with a least-squares regression line $y = mk + b$ for time index k.

which is better adapted to the trend than a constant server rate C, if the goal is a fixed utilization ρ.

Figure 20 shows queue length CCDF plots for queueing simulations using the original and replayed data, an infinite size buffer, and the variable server rate based on the linear regression line. Two utilizations are shown: 65% (left panel) and 85% (right panel). It can easily be seen that the queue lengths are much smaller from the replayed data.
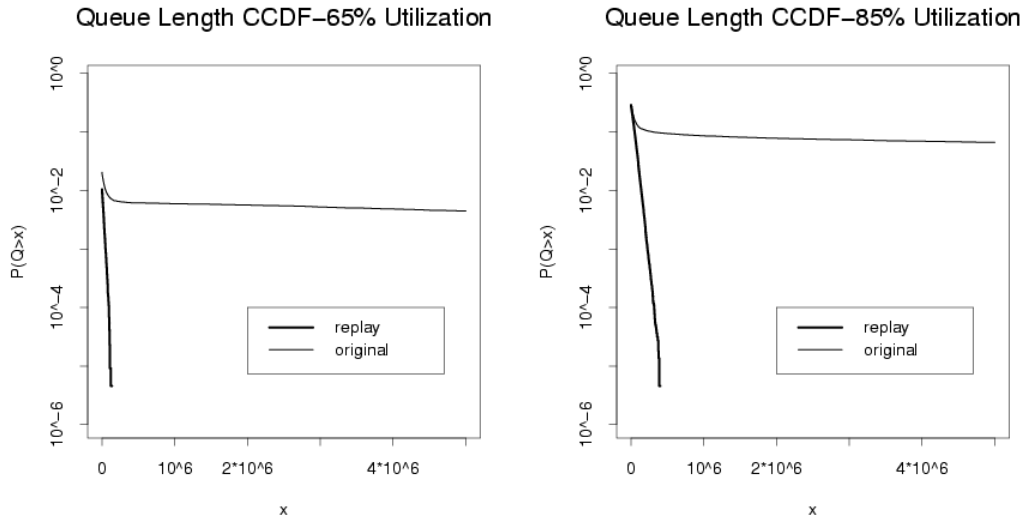

Fig. 20. Comparison of empirical queue length CCDF for original and replayed data, at 65% utilization (left panel) and at 85% utilization (right panel). The tail of the CCDF from the replayed data (thick line) decays very quickly compared with the original data (thin line). The server rate was adapted using a mean trend line.

A second approach to address the upward trend offers more flexibility than provided by a regression line. 'Local means' are calculated on non-overlapping

24

intervals of fixed length. Then, on each such interval the local mean and desired utilization are used to calculate the server rate. In other words, the server rate is allowed to vary as the local means vary. This provides a way to adapt to changing mean rates, a kind of non-stationarity.

Using a queueing simulation with a server rate that adapts by the local means, the loss processes further demonstrate the fundamental differences in the characteristics of the original and replayed traces, as shown in Figure 21. The service rate used corresponded to a block mean process estimated at 1 sec intervals. It can be seen that the loss process for the original data is significantly more 'spiky' compared to that of the replayed data.
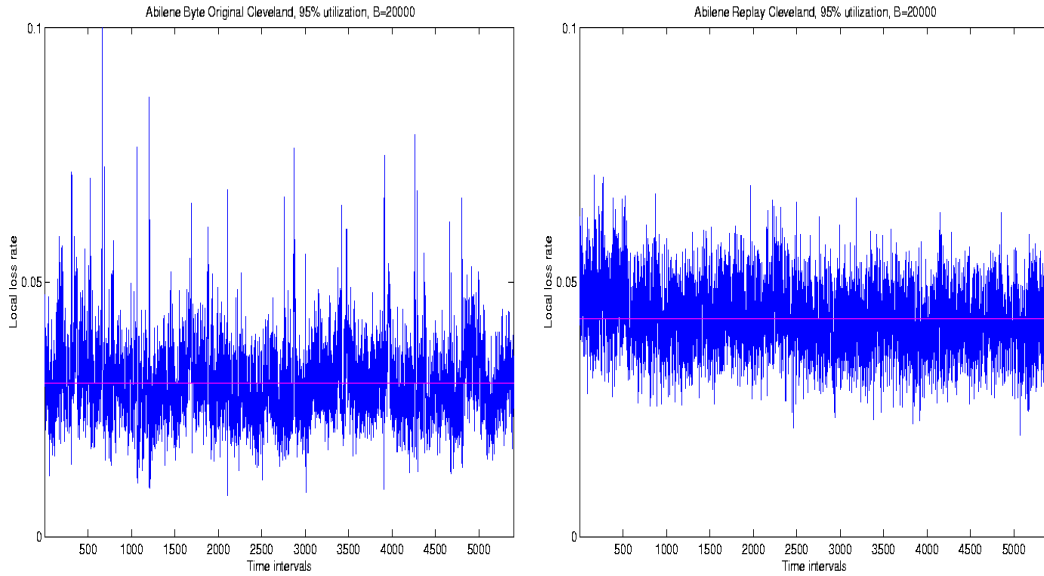

Fig. 21. Aggregated loss rate processes for the original (left panel) and replayed (right panel) traces at 95% utilization and with a buffer of size 20000.

Figure 22 shows a multiscale excursion map, together with the input process, for the original Abilene Cleveland byte data. For this plot the threshold levels were set to 200000 and 2 million bytes, respectively. In this plot, we see again long-lasting excursions above the 2 million byte level at low levels. However, this is mainly due to the non-stationary nature of the input process, since when the service rates become more adaptive (at higher levels), the excursions of the workload process become significantly smaller. It is worth noting that even at higher levels (when the service rate process has become fairly adaptive to local fluctuations) the excursions have longer durations than those observed for the FGN trace, thus indicating a stronger temporal dependence.

The conclusion of this trace-driven queueing analysis provides substantial evidence that the original and replayed traces appear quite different from a queueing point of view. The replayed trace is smoother, and therefore less queueing intensive. The queue is typically much smaller using the replayed
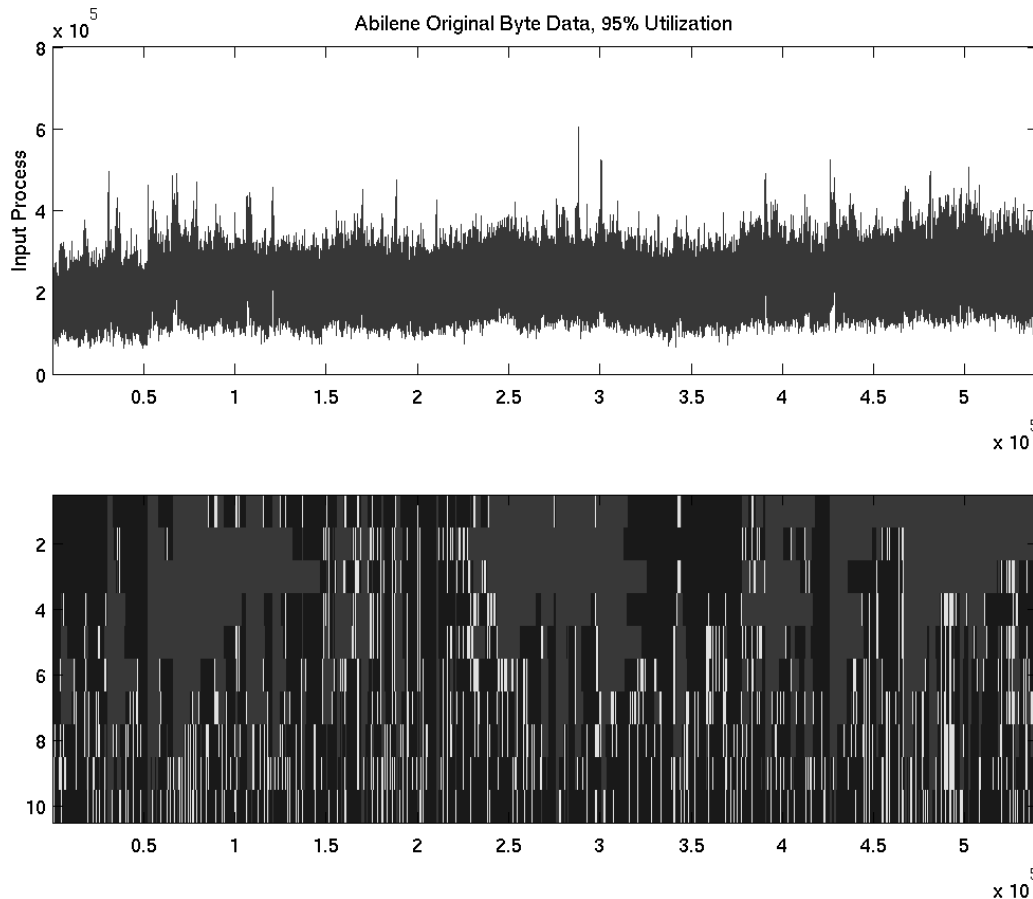
Fig. 22. Abilene trace (top) and corresponding multiscale excursion map (bottom) for values <200,000 (black), 200,000-2,000,000 (white), and >2,000,000 (grey).

trace, and with a finite buffer the loss process is also much smoother. It is not clear that research results obtained with the replayed data would also hold for the original traffic which has far more variability and is more queueing intensive. In this case, the differences in the traces were important.

A more thorough trace-driven queueing analysis of the original and replayed Abilene Cleveland data using an infinite buffer is accessible via the Internet [46]. The mean trend is handled in three ways: ignoring it, adapting the server rate using the linear regression line, and adapting the server rate using the 1 second local means. An additional original/replay combination, Abilene Indianapolis, is also compared. For both combinations, trace-driven queueing analysis generally supports the conclusion that the replay data is quite different from the original traces, from a queueing standpoint. Through a separate analysis (not shown), differences in the dependence structure of the two datasets account for almost all of the differences in the queueing behavior.

Following this queueing analysis, further study revealed that, while the source-level representation of the connections in the trace was accurate enough, the

use of *common sense* values for some network-dependent parameters in the experiment was the major cause of the difference. In particular the replay trace made used of round-trip times normally distributed between 40 and 150 milliseconds (matching delays in the continental U.S.), a fixed TCP receiver window size (17,520 bytes) and no capacity constraints (all connection could potentially reach 100 Mbps). Our work demonstrated that it is important to measure not only source-level properties but also network-dependent ones in order to obtain accurate replays.

As part of the network experiments, measurement tools have been expanded to provide a characterization of the distributions of round-trip times, windows sizes and capacities of the connections in the input packet header trace. For the Abilene trace considered above, we found that round-trip times are log-normally distributed, with most values between 30 and 300 millisecond, with a significantly long tail. This property, combined with an observed wide range of window sizes, contributes to increase the burstiness of the packet arrival process at fine time-scales more substantially that expected. After we incorporated these distributions in our replay experiment, the generated traffic resembled the original trace more closely. We have demonstrated that accurate source-level characterizations can reproduce the long-range dependence in the original traffic, but they are unable to reproduce the finer structure of the packet arrival process *unless* the primary network-dependent parameters are also considered in the experiment.

Figure 23 shows preliminary results for a more recent version of the UNC replay tool. In this case the original byte count data was taken from the outbound link from UNC to the Internet. The complete original and replay traces contained start-up and end effects that prevented using entire traces as stationary sequences. So, for this comparison a stationary middle portion of the data with length 110,000 was isolated from both the original and replay traces. Because these traces are so short, the conclusions must be taken as preliminary. Figure 23 (left panel) shows the empirical queue length CCDF with a utilization of 70%. The curves appear remarkably similar, which shows that the tool seems to be capturing well features of the data relevant for queueing. This graph is typical of those for utilizations from 60% to 70%. (Below about 55% there is no queueing.) Interestingly, at 75% utilization and above the graphs show a different situation (right panel). For these higher utilizations a feature in the replay data leads to larger queues. The queue length time plot (Figure 24- left panel) shows a significant spike around time index 72,000 caused by a "burst and bump" in the trace (Figure 24- right panel) . For a small time, even the smallest bin counts are larger than usual. The queue doesn't have a chance to catch up to the increased workload in this feature. The tool developers must now go back and understand why this feature appears in their replay but not the original trace.
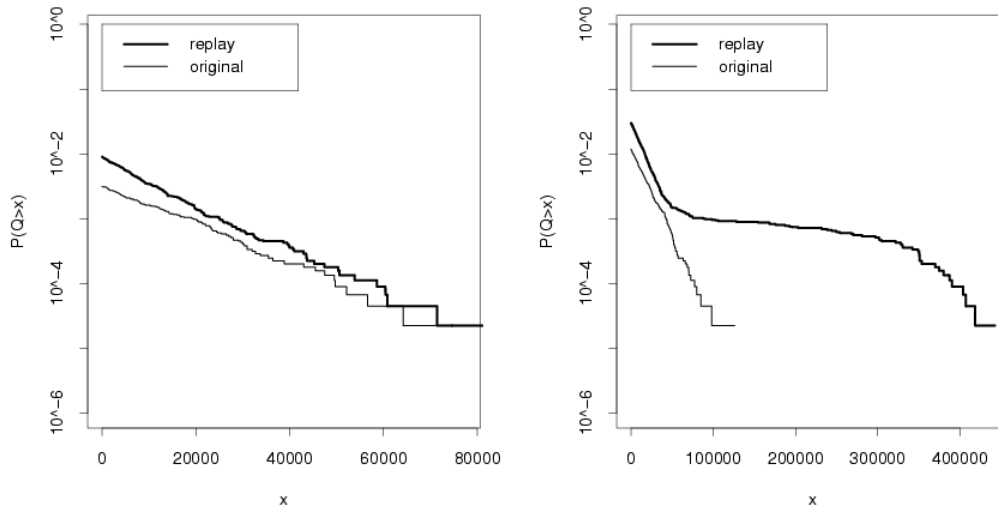
Fig. 23. With a newer version of the UNC tool, the queue length distribution of original and replayed data is quite similar for utilizations 70% and below, but different at 75% and above.
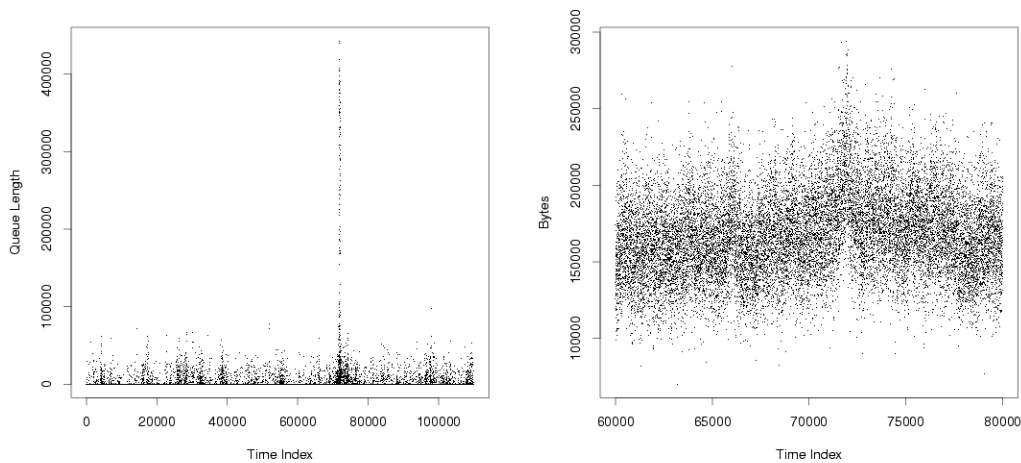

Fig. 24. The queue length time plot (left) allows localizing the large queue to around time index 72,000, caused by a noticeable "burst and bump" in the replayed trace around time index 72,000.

5 Problems and Issues

5.1 *Statistical Measures Don't Easily Translate*

Queues can be very sensitive to the marginal distribution and dependence structure of the input data. Taralp, Devetsikiotis and Lambadaris [49] demonstrated this in a simulation study. They generated three different non-Gaussian datasets using different models but with virtually identical marginal distributions and autocorrelations. Fed to a simulated server, the queueing results

28

were quite different. Obviously the autocorrelation does not capture the entire dependence structure, but doing so is very difficult in general. Modeling the dependence structure sufficiently well to understand the queueing behavior is difficult. It also begs the question of knowing when the model is sufficiently good to understand the queueing.

5.2 Non-Gaussian Data

Real network data presents a number of complications for queueing simulation and analysis. Figure 25 (left) shows the byte counts for inbound traffic to UNC measured on April 9, 2002 at 8 AM in 10 ms intervals. The data shows "spikes" that would not be captured by a marginal Gaussian distribution and an upward trend consistent with increasing activity as the morning progresses. Non-Gaussianity is not unusual in 10 ms data. If one aggregates to 1s or even 10s the data will appear more Gaussian but one must question whether the longer timescales are relevant. As described in [43] a Gaussian assumption can also lead to over-optimistic predictions of tail-queue probability.
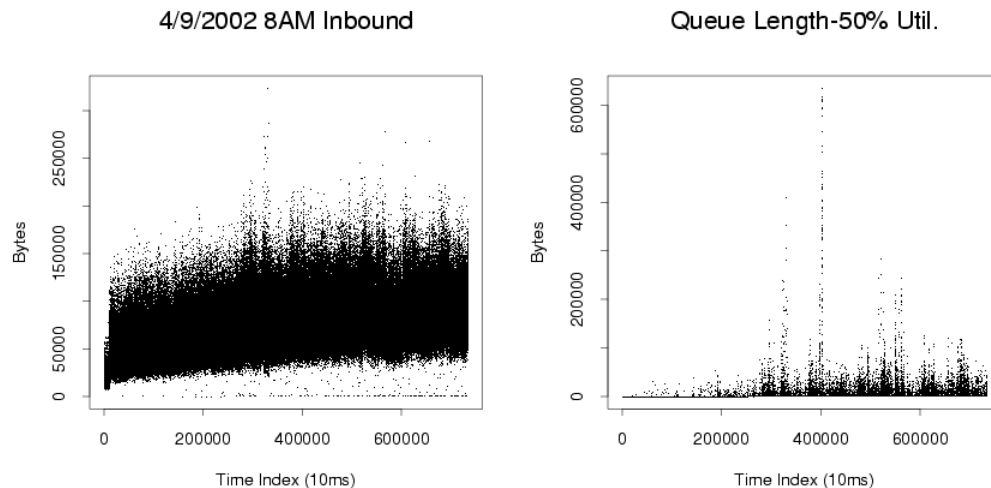

Fig. 25. Byte counts of UNC inbound traffic (left) and queue length at 50% utilization. The increasing trend in the traffic is reflected in the increasing queue lengths.

5.3 Mean Trends and Shifts

As mentioned in Section 4, trends in the input process speak against the use of stationary models. Using the sample mean of the entire trace to determine the server rate results in less queueing at the start of the trace and more queueing at the end of the trace (Figure 25 - right). A "steady-state" queue length distribution wouldn't be meaningful. A possibility is to adapt the server
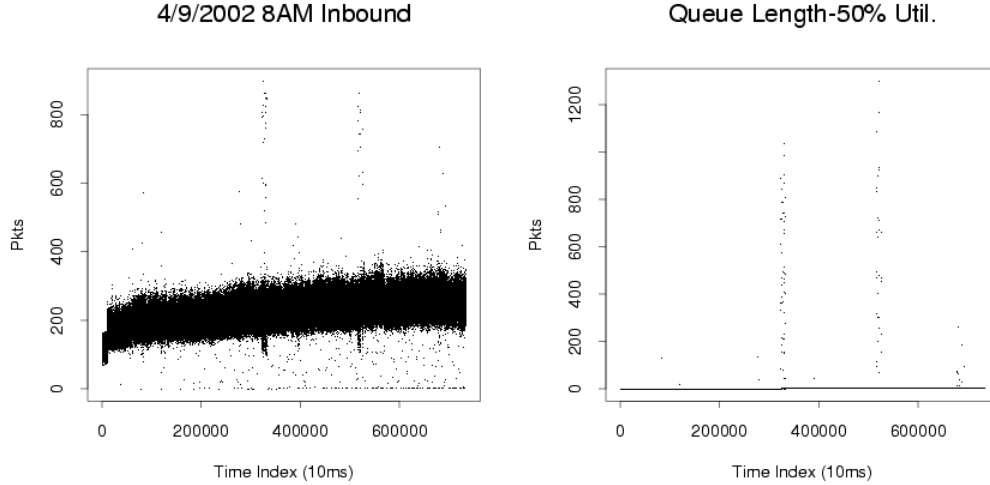
29

Fig. 26. Packet counts of UNC inbound traffic (left) and queue length at 50% utilization. Spikes are more prominent than in the corresponding byte counts (see Figure 25).

rate to the increasing mean. This has a loose physical interpretation in which increasing bandwidth is given to the source as traffic demand increases. This is really an analogue to the practice of "de-trending", but here the data is left unchanged.

5.4   Utilization Doesn't Scale Like the Data

Let $Y = \{Y_i,\ i = 1, \ldots, n\}$ be a stationary input sequence with finite mean μ. The block means process is

$$Y_j^{(m)} := \frac{1}{m} \sum_{i=(j-1)m+1}^{jm} Y_i,\ k = 1, \ldots, \lfloor N/m \rfloor. \tag{9}$$

A defining property of Fractional Gaussian Noise is self-similarity, for which $Y^{(m)} \overset{D}{=} m^{H-1} Y$ (where $\overset{D}{=}$ denotes equality in the sense of the finite-dimensional distributions). For example, if Y is bytes per millisecond, $Y^{(10)}$ is the average bytes per 10 ms, $Y^{(100)}$ is the average bytes per 100 ms and so on. (Similarly $mY^{(m)}$ gives the block sums.) However, assuming Y is an infinite sequence and satisfies a Law of Large Numbers, $Y_j^{(m)} \to \mu$ as $m \to \infty$.

Figure 27 (left column) shows the block means of the same Fractional Gaussian Noise shown in Figure 6 (right) for aggregation levels $m = 100$, 1,000, and 10,000. In the left column the y-axis is naturally scaled by the smallest and largest values in the data. Such graphs have been given as "pictorial proof" of self-similarity [31, Figure 4]. Figure 27 (right column) shows the same block means data but with the y-axis limits fixed to those used in Figure 6 (right).
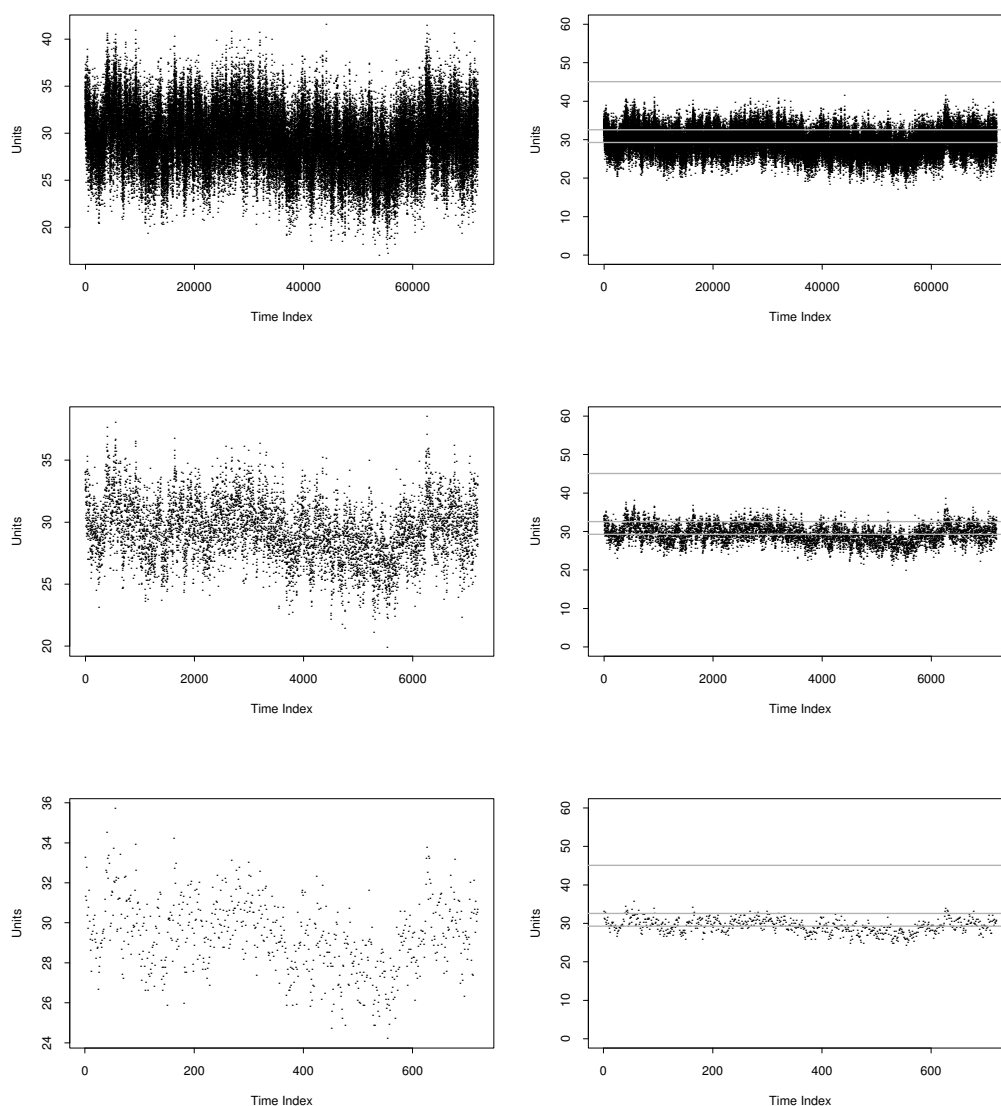
30

Fig. 27. Block means of Fractional Gaussian Noise with aggregation levels (top to bottom) $m = 100$, 1,000, and 10,000. Horizontal lines correspond to (bottom to top) the sample mean, the server rate for 90% utilization, and the server rate for 65% utilization. The left column allows the y-axis to change with the data while the right column has fixed y-axis limits. While the data exhibits self-similarity (left), the Law of Large Numbers brings the block means closer to the overall mean (right). Even at 90% utilization there would be very little queueing if data were aggregated by a factor of 10,000. Compare with Figure 6.

With increasing aggregation level m the data shows less variability around the mean. As a consequence, server rates corresponding to 65% or 90% utilization in the original data would result in much lower utilizations with large aggregation level m. For example, the lower-left graph of Figure 27 suggests that using the server rate that gives 90% utilization in the original data results in

practically no queueing when the data is averaged with block size $m = 10,000$.

5.5   Packet Counts vs Byte Counts

Figure 26 (left) shows packet counts for the same UNC data shown in Figure 25. Packets are distributed with sizes between about 36 bytes and 1500 bytes. But, depending on the packet sizes, a spike in the packet count may or may not correspond to a spike in the byte count. Although either byte or packet counts can be used as input traffic to a simulated server, byte counts are more natural. In hardware, bandwidth is in bytes (actually bits) per unit time and so it is more natural to simulate a queue that drains in the same units. Otherwise one is implicitly assuming all packets are roughly the same size. In practice IP packets can easily range from 40 bytes (a packet with minimal payload) to 1500 bytes (the maximum size of an Ethernet frame).

6   Concluding Remarks

The motivation for the present work originated from the difficulties presented by using classical statistical measures to capture the intricate differences between real network traffic traces and their synthetic counterparts. It is shown that trace-driven queueing can capture their temporal characteristics and quantify the effect of bursts successfully.

Queueing of traffic traces depends on the utilization rate (for the infinite buffer case) and on the utilization rate and the buffer size for the finite case. In order to gain insights into the workings of queueing a number of metrics are introduced together with the corresponding visualization tools.

Using our proposed queueing framework we examine how synthetically generated traces from the UNC group compare to the original versions and note which features have been replicated. Finally, we outline for users of our framework several issues related to trace-driven queueing.

Non-stationary behavior is not uncommon in network traffic traces. There remains a need for additional theory and tools to enable trace-driven queueing comparisons under more general non-stationary conditions.

Acknowledgements

References

[1] P. Abry and D. Veitch. Wavelet analysis of long-range dependent traffic. *IEEE Trans. on Information Theory*, vol. 44, no. 1, pp. 2–15, Jan. 1998.

[2] M. Allman, V. Paxson, and W. Stevens. *RFC 2581: TCP Congestion Control*, April 1999.

[3] S. Asmussen. *Applied Probability and Queues*, Wiley, New York, 1987.

[4] P. Bremaud and F. Baccelli. *Elements of Queueing Theory*, Springer Verlag, 2002.

[5] J. Cao, W. Cleveland, D. Lin, and D. Sun. Internet traffic tends *toward* Poisson and independent as the load increases. *Nonlinear Estimation and Classification*, C. Holmes, D. Denison, M. Hansen, B. Yu, and B. Mallick, eds., Springer, New York, 2002.

[6] O. Cappé, E. Moulines, J. Pesquet, A. Petropulu, and X. Yang. Long-range dependence and heavy-tail modeling for teletraffic data. *IEEE Signal Processing Magazine*, 19:14–27, May 2002.

[7] P. Doukhan, G. Oppenheim and M. S. Taqqu, eds. *Theory and Applications of Long-Range Dependence*. Birkhäuser, Boston, 2003.

[8] N.G. Duffield and N. O'Connell. Large deviations and overflow probabilities for the general single-server queue, with applications. *Math Proc. Camb. Phil. Soc*, vol. 118, pp. 363–374, 1995.

[9] A. I. Elwalid and D. Mitra. Effective bandwidth of general Markovian traffic sources and admission control of high speed networks. *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 329–343, 1993.

[10] Endance Measurement Systems, DAG 4.2 OC-48 Network Measurement Card, `http://www.endace.com/dag4.2.htm`.

[11] A. Erramilli, O. Narayan and W. Willinger. Experimental queueing analysis with long-range dependent traffic. *IEEE/ACM Trans. on Networking*, vol. 4, pp. 209–223, April 1996.

[12] W. Feller. *An Introduction to Probability Theory and Its Applications* vol. 2, 2nd ed., Wiley, New York, 1971.

[13] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[14] S. Floyd and V. Paxson, Difficulties in Simulating the Internet, *IEEE/ACM Trans. on Networking*, vol. 9, no. 4, pp. 392–403, 2001.

[15] C. Fraleigh, F. Tobagi, and C. Diot. Provisioning IP backbone networks to support latency sensitive traffic. In *Proc. IEEE INFOCOM 2003*, April 2003.

[16] P. W. Glynn and W. Whitt. Logarithmic asymptotics for steady-state tail probabilities in a single-server queue. *Journal of Applied Probability*, pp. 131–155, 1994.

[17] A. Hussain, J. Heidemann, and C. Papadopoulos. A framework for classifying denial of service attacks. In *Proc. ACM SIGCOMM*, Germany, August 2003.

[18] F. Hernandez-Campos, K. Jeffay, and F. Donelson Smith. Modeling and Generating TCP Application Workloads. In submission, July 2004.

[19] F. Hernandez-Campos, L. Le, J. S. Marron, C. Park, J. Park, V. Pipiras, F. D. Smith, R. L. Smith, M. Trovero, and Z. Zhu. Long range dependence analysis of Internet traffic. In preparation, 2004.

[20] F. Hernandez-Campos, F. Donelson Smith, and K. Jeffay. How "Real" Can Synthetic Network Traffic Be? (poster abstract) In Proceedings of ACM SIGCOMM'04, August 2004.

[21] C. V. Hollot, V. Misra, D. F. Towsley, and W. Gong. On designing improved controllers for AQM routers supporting TCP flows. In *Proc. IEEE Infocom*, pp. 1726–1734, 2001.

[22] J. Hüsler and V. Piterbarg, Extremes of a Certain Class of Gaussian Processes, *Stochastic Processes and their Applications*, vol. 83, pp. 257–271, 1999.

[23] V. Jacobson. Congestion avoidance and control. In *Proceedings of ACM SIGCOMM*, Stanford, CA, August, 1988.

[24] S. Jamin, P. B. Danzig, S. J. Shenker, and L. Zhang. A measurement-based admission control algorithm for integrated service packet networks. *IEEE/ACM Trans. on Networking*, vol. 5, no. 1, pp. 56–70, 1997.

[25] Y. Joo, V. Ribeiro, A. Feldmann, A.C. Gilbert, and W. Willinger. TCP/IP traffic dynamics and network performance: A lesson in workload modeling, flow control, and trace-driven simulations. *Computer Communication Review*, vol. 31, pp. 25–37, 2001.

[26] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido. A nonstationary Poisson view of internet traffic. In *Proc. IEEE INFOCOM*, Hong Kong, March, 2004.

[27] A. Karasaridis and D. Hatzinakos. Network heavy traffic modeling using stable self-similar processes. *IEEE Transactions in Communications*, vol. 49. no. 7, pp. 1203–1214, July 2001.

[28] S. Karlin and H. M. Taylor, *A First Course in Stochastic Processes*, 2nd. ed., Academic Press, New York, 1975.

[29] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. *Computer Communication Review*, vol. 32, no. 4, pp. 89–102, 2002.

[30] G. Kesidis, J. Walrand, and C.-S. Chang. Effective bandwidths for multiclass Markov fluids and other ATM sources. *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 424–428, 1993.

[31] W. E. Leland, M. S. Taqqu, W. Willinger and D. V. Wilson. On the self-similar nature of Ethernet traffic (Extended Version). *IEEE/ACM Trans. on Networking*, vol. 2, no. 1, pp. 1–15, 1994.

[32] S. H. Low, F. Paganini and J. C. Doyle. Internet Congestion Control, *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 28–43, 2002.

[33] L. Massoulie and A. Simonian. Large buffer asymptotics for the queue with fractional Brownian Input. *Journal of Applied Probability*, vol. 36, pp. 894–906, 1999.

[34] T. Mikosch, S. Resnick, H. Rootzén, and A. Stegeman. Is network traffic approximated by stable Lévy motion or Fractional Brownian motion? *Ann. Appl. Probab.*, vol. 12, no. 1, pp. 23–68, 2002

[35] National Laboratory for Advanced Network Research, `http://moat.nlanr.net/Images/Navbar/2_MNA.html`.

[36] I. Norros. A storage model with self-similar input. *Queueing Systems*, vol. 16, pp. 387–396, 1994.

[37] I. Norros. On the use of Fractional Brownian Motion in the theory of connectionless networks. *IEEE J. Select. Areas. Commun.*, vol. 13, pp. 953–962, Aug. 1995.

[38] I. Norros. Queueing behavior under Fractional Brownian traffic. In *Self-similar Network Traffic and Performance Evaluation*, K. Park and W. Willinger, eds., pp. 101–114, John Wiley & Sons, New York, 2000.

[39] K. Park and W. Willinger, eds., Self-Similar Network Traffic and Performance Evaluation, Wiley Interscience, 2000.

[40] V. Paxson. Fast, approximate synthesis of Fractional Gaussian Noise for generating self-similar network traffic, *Computer Communication Review*, vol. 27, no. 5, pp. 5–18.

[41] V. Paxson and S. Floyd. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Trans. on Networking*, vol. 3, no. 3, pp. 226–244, 1995.

[42] J. Postel. *RFC 793 (STD 7): Transmission Control Protocol*, September 1, 1981.

[43] V. Ribeiro, R. Riedi, M. Crouse, R. Baraniuk. Multiscale Queuing Analysis of Long-Range Dependent Network Traffic. *Proc. IEEE INFOCOM'00*, Tel Aviv, Israel, 2000.

[44] L. Rizzo. Dummynet: A simple approach to the evaluation of network protocols, *Computer Communication Review*, vol. 27, No. 1, January 1997, pp. 31–41

[45] D. A. Rolls. Limit Theorems and Estimation for Structural and Aggregate Teletraffic Models, Ph.D. thesis, Queen's University at Kingston, 2003.

[46] D. A. Rolls. Queueing Comparison of Original and Replayed Abilene Data, 2003. (Unpublished) URL: `http://www.samsi.info/200304/int/work/testbed/abilene1/queue_abiline1.html`

[47] M. S. Taqqu, W. Willinger and R. Sherman. Proof of a fundamental result in self-similar traffic modeling. *Computer Communication Review*, vol. 27, pp. 5–23, 1997.

[48] M. S. Taqqu. The modelling of Ethernet data and of signals that are heavy-tailed with infinite variance. *Scand. J. Statist.*, vol. 29, no. 2, pp. 273–295, 2002.

[49] T. Taralp, M. Devetsikiotis, and I. Lambadaris. In search of better statistics for traffic characterization. *Journal of the Brazilian Computer Society*, vol. 5, no. 3, pp. 5–13, 1999.

[50] D. Veitch, and P. Abry. Code for the estimation of scaling exponents, `http://www.emulab.ee.mu.oz.au/~darryl/secondorder_code.html`, 2002.

[51] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Trans. on Networking*, vol. 5, no. 1, pp. 71–86, 1997.

[52] Z.-L. Zhang, V. Ribeiro, S. Moon, and C. Diot. Small-time scaling behaviors of Internet backbone traffic: an empirical study. In *Proc. IEEE Infocom*, San Francisco, March, 2003.