Self-Similarity in File Systems

Steven D. Gribble, Gurmeet Singh Manku, Drew Roselli, and Eric A. Brewer
Computer Science Division, University of California at Berkeley
{gribble,manku,drew,brewer}@cs.berkeley.edu

Timothy J. Gibson and Ethan L. Miller
Computer Science and Electrical Engineering Department,
University of Maryland Baltimore County
{tgibso2,elm}@cs.umbc.edu

Abstract

We demonstrate that high-level file system events exhibit self-similar behaviour, but only for short-term time scales of approximately under a day. We do so through the analysis of four sets of traces that span time scales of milliseconds through months, and that differ in the trace collection method, the filesystems being traced, and the chronological times of the tracing. Two sets of detailed, short-term file system trace data are analyzed; both are shown to have self-similar like behaviour, with consistent Hurst parameters (a measure of self-similarity) for all file system traffic as well as individual classes of file system events. Long-term file system trace data is then analyzed, and we discover that the traces' high variability and self-similar behaviour does not persist across time scales of days, weeks, and months. Using the short-term trace data, we show that sources of file system traffic exhibit ON/OFF source behaviour, which is characterized by highly variably lengthed bursts of activity, followed by similarly variably lengthed periods of inactivity. This ON/OFF behaviour is used to motivate a simple technique for synthesizing a stream of events that exhibit the same self-similar short-term behaviour as was observed in the file system traces.

1 Introduction

Recent studies of high quality traces of network traffic (see [9], [2], and [15]) have revealed an unexpected property of network traffic, namely that the traffic appears to be *self-similar* in nature. Intuitively, a self-similar process looks similar and bursty across all time-scales. Of course, physical contraints (such as bandwidth limits and the finite lifetime of any given network) mean that no real network can ever be truly self-similar — but the important properties of self-similar systems do manifest themselves across many time scales that are of practical importance.

The theoretical and practical consequences of having a self-similar process cannot be dismissed. A common assumption in the design of networks and operating systems is that the aggregation of a large number of bursty sources tends to be smooth. For example, statistical admissions criteria in real-time channel establishment [5] assume that while an individual source may exceed its average resource requirements at any given time, the aggregate resource requirements across many such sources has a low variance. If the combined traffic is self-similar, this is not necessarily the case.

Further studies have shown that the total traffic (measured in bytes/s or packets/s) on Ethernet LANs [9] and on WANs [16] is self-similar. Similarly, investigations into variable-bit-rate (VBR) video traffic [2] have shown this traffic to exhibit long-range dependence, which is an indicator of self-similarity. The purpose of this paper is to demonstrate the existence of self-similar behaviour in high-level file system events (such as file opens, block reads or writes, file closes, and file deletions) over short-term time scales of less than a day in length, but to show that this behaviour does not persist across time scales of days, weeks, and months. We refer to this property as "short term self-similarity": a process possesses short-term self-similarity if it is well represented as a self-similar process over short time scales, but not over larger time scales. Short term self-similarity is nearly an oxymoron in the sense that it is strictly **not** self-similarity; for a process to be truly self-similar, its self-similar behaviour must extend through all time scales.

Our contributions in this paper are:

- to demonstrate that four different file systems exhibit short term self-similarity through the analysis of traces of those systems' activity,
- to explicitly show using two of those file system traces that the self-similar behaviour of the file systems breaks down past day, week, and month-long time scales,
- to show that over short time scales, file system activity is composed of highly variable ON/OFF activity from a number of clients, which may explain the short term self-similarity,
- and to present a model for synthesizing file system traffic that exhibits self-similar behaviour.

This paper is structured as follows. Section 2 discusses related work. In section 3, the theory behind self-similar and long-range dependent processes is summarized. Section 4 describes the four file system traces analyzed in this paper. Section 5 contains the analysis of Sprite LFS and Auspex NFS short-term file system traces, which exhibit self-similar behaviour. In section 6, a similar analysis is performed on two sets of much longer time scale data (from an instructional file system at U.C. Berkeley, and from an NFS file system at the University of Maryland Baltimore Country), which reveals that this self-similar nature does not persist across these longer time scales. Finally, section 7 shows that the short-term trace data exhibits highly variable ON/OFF sources, which is used to motivate a method for synthesizing self-similar data. A summary of this paper is presented and general conclusions drawn in section 8.

2 Related Work

The study of self-similarity in computer networks was pioneered by the work of Leland et al., in which they demonstrated that Ethernet traffic was self-similar in nature [9].[1] Further work showed that the self-similarity could be attributed to the ON/OFF behaviour of traffic sources within their system [23]. We borrow heavily upon the theory and analysis techniques presented in these two papers to demonstrate the presence of self-similarity in file system traffic. Self-similar behaviour in various other types of systems (such as wide-area traffic [16], ATM networks [6], variable-bit-rate video [2], and World Wide Web traffic [3]) has been detected using similar techniques.

The analysis of file system performance, access characteristics, and traffic patterns has received considerable attention in the past few years. In [19], the effects of file layout and fragmentation of a disk on file system performance are measured using synthesized workloads. Baker et al. analyzed the user-level file access patterns in the Sprite distributed file system using traces gathered via kernel instrumentation in a Sprite installation [7, 1]. The design and performance of a log-structured file system is presented in [18], including the analysis of various cleaning policies. The HP AutoRAID hierarchical storage system is discussed in [22].

3 Theory of Self-Similarity

The theory behind self-similar processes is briefly presented in this section. A more thorough treatment can be found in [9], [23], or [16]; the goal of this section is to outline enough of the theory to motivate the methodology discussed in section 5.

Consider a stochastic process $X = (X_1, X_2, \ldots)$ with mean $\mu = E\{X_i\}$, variance $\sigma^2 = E\{(X_i - \mu)^2\}$, and autocorrelation function:

$$r(k) = \frac{E\{(X_i - \mu)(X_{i+k} - \mu)\}}{E\{(X_i - \mu)^2\}}, \quad \text{for } k \geq 0. \tag{1}$$

This process is said to exhibit *long-range dependence* if

$$r(k) \sim k^{-\beta} L(t) \;\; \text{for} \;\; (0 < \beta < 2), \;\; \text{as } k \to \infty \tag{2}$$

and $L(t)$ is a slowly varying function with the property

$$\lim_{t \to \infty} L(tx)/L(t) = 1 \;\; \forall\, x > 0. \tag{3}$$

A new aggregated time series $X^{(m)} = (X_k^{(m)} : k = 1, 2, 3, \ldots)$ for each $m = 1, 2, 3, \ldots$ is obtained by averaging non-overlapping blocks of size m from the original series X. In other words,

$$X_k^{(m)} = \frac{(X_{km-m+1} + \cdots + X_{km})}{m}, \quad k \geq 1. \tag{4}$$

[1] Since most Ethernet traffic is NFS traffic, it is not unreasonable to expect that NFS traffic shows behaviour akin to the Ethernet traffic.

The process X is said to be *exactly second-order self-similar* with Hurst parameter

$$H = 1 - \frac{\beta}{2} \;\; (0 < \beta < 2) \tag{5}$$

if, for any $m = 1, 2, 3, \ldots$,

$$Var\left(X^{(m)}\right) \propto m^{-\beta} \tag{6}$$

and

$$r^{(m)}(k) = r(k) \quad (k = 1, 2, 3, \ldots). \tag{7}$$

If the weaker condition

$$r^{(m)}(k) \to r(k) \;\; \text{as } m \to \infty \tag{8}$$

is true, then the process X is said to be *asymptotically second-order self-similar*, with Hurst parameter H again given by equation 5.

As described in [9], self-similar processes provide an explanation for an empirical law known as the *Hurst effect.* The *rescaled adjusted range (R/S) statistic* for a set of observations $(X_k : k = 1, 2, \ldots, n)$ having mean $\bar{X}(n)$ and sample variance $S^2(n)$ is given by the relation

$$\begin{aligned} R(n)/S(n) = \;& (1/S(n))[\max(0, W_1, W_2, \ldots, W_n) - \\ & \min(0, W_1, W_2, \ldots, W_n)] \end{aligned} \tag{9}$$

where

$$W_k = (X_1 + X_2 + \cdots + X_k) - k\bar{X}(n) \;\; (k \geq 1). \tag{10}$$

Short-range dependent sets of observations seem to satisfy $E[R(n)/S(n)] \sim c_0 n^{\frac{1}{2}}$, while long-range dependent processes such as self-similar process are observed to follow

$$E\left[\frac{R(n)}{S(n)}\right] \sim c_0 n^H, \;\; (0 < H < 1). \tag{11}$$

This is known as the Hurst effect, and it can be used to differentiate between self-similar and non-self-similar time series, as will be demonstrated in section 5.1.[2]

4 File System Traces

We examine a total of four file system traces in this paper. The first set of traces was collected in 1991 for a study of the file access patterns and caching behaviour of the Sprite distributed file system [1]. The traces were collected on a Sprite cluster of approximately 40 workstations sharing a single Ethernet, over eight separate 24 hour intervals. Because of the nature of the Sprite distributed file system and the manner in which the traces were gathered, all file system events required communication between the client and the server; thus, all file system events could be gathered at the servers themselves. The events within the file trace were collected through kernel call instrumentation on client machines with a precision of 10 milliseconds. All file system

[2] The Hurst parameter can be broken down into three distinct categories; $H < 0.5$, $H = 0.5$, and $H > 0.5$. Ordinary Brownian motion (e.g., a random walk) is produced when $H = 0.5$. If $H > 0.5$, the values produced are self-similar with a positive correlation. If $H < 0.5$, the values produced are self-similar with a negative correlation. Negative correlated data tends to reverse itself instead of continuing along the same path. Positive and negative correlated data is also called persistent and anti-persistent data, respectively. [12] Most observed self-similar data to date is persistent.

requests (including those that are satisfied by a client-side cache) are present within the trace. The Sprite traces suffer from some minor limitations, such as a lack of fully complete information on file read and write events. The total number of bytes read from and written to files could be deduced from the seek and close events recorded in the trace. However, each individual file read and write event could not be recovered — these events were recorded in our second set of traces.

The second set of traces used in this paper was gathered from a relatively large NFS installation served by a single Auspex NFS server. These traces were collected in 1994 to study the impact of different cache policies on scalable network file system performance [4]. The Auspex NFS server straddled four separate Ethernets, and served a total of 237 clients. The traces were collected by monitoring network activity on each of the four Ethernets. This implies that any file system request satisfied by a client-side NFS cache was not present in the trace. The NFS traces available to us had been post-processed in order to remove NFS-related artifacts and deficiencies. For example, file open and close events were not visible from the network, and had to be added to the traces using a heuristic method. Postprocessing was also performed on NFS attribute operations; it was assumed that most attribute read operations within the trace were really NFS cache consistency validation operations, and thus some were removed from the trace. Anomalies from this postprocessing likely introduced some minor distortion into our results, but this distortion is only on a very fine time scale (on the order of milliseconds), and the fact that the statistical methods used to detect self-similarity take into account many time scales (up to many hours) implies that the effects of such distortion are minimal.

The third set of traces was collected on a cluster of twenty instructional UNIX workstations used by undergraduate classes at UC Berkeley [17]. These long term traces (referred to as the INS traces throughout the paper) were gathered by using the auditing system to log all system calls relevant to the file system on each workstation. Because the traces were collected on the client workstations, requests that are absorbed by the local cache are present in the traces. File system calls recorded in the raw traces include individual data operations, such as read and write requests, as well as metadata operations, such as open, stat, and chmod. However, in the processed version of these traces that is analyzed in this paper, some calls (such as lseek and dup) have been removed from the traces, and consecutive read or write operations made by the same process during the same second are coalesced. Time is recorded on the granularity of one second. These traces were collected continuously for six and one half months over the fall of 1996 and spring of 1997.

Event Type	#Sprite Events	# NFS Events R	W	RW	# INS Events
Open	38668	66845	8457	520	229968287
Close	48412	66714	8455	518	232488430
Block Xfer	n/a	344564	71399	0	633158006
Delete	5764	0	3327	0	3147359
Attribute	unused	1374402	144336	0	857738997
Directory	unused	1890262	0	20201	32176538
Seeks	28371	n/a			n/a
Total #:	121215	4000000			1988677617
Time:	14413s (4h)	91337s (25.37h)			17448960s (6 mnth)

Table 1: **Trace summaries:** this table summarizes the number and type of events available in the Sprite, Auspex NFS, and INS traces.

Trace Attribute	**Value**
FS type	NFS
FS size	7.5 GB
# files	250,000
# users	500
# workstation clients	150
Trace duration	287 days
Trace start	October 23rd, 1996
# Creations	443,516 (1,545/day)
# Deletions	314,333 (1,100/day)
# File Accesses	807,968 (2,815/day)
# Modifications	108,262 (377/day)

Table 2: **Summary of the UMBC trace data**

The final set of traces were collected daily on a portion of the University of Maryland Baltimore County's file systems to generate data for a tape-migration simulation. The tracing period spanned 287 days. These traces (referred to as the UMBC traces) were not collected with kernel modifications; instead, a modified version of the "find" utility was run nightly to collect information from a number of file systems in the University. This tracing process did not provide the fine-grained data the Sprite and NFS traces possess — the tape-migration simulator did not require such fine-grained measurements. For example, on the Sprite system traces, a user modifying a file will generate an open, a close, and many intervening reads and writes. The UMBC traces only show one action, a file modification. Thus, the long-term traces collapse many events into one. Similarly, the trace collection process does not track how many times a file is used during a day. Finally, it misses all the temporary files the system creates and deletes during the day — Ousterhout [14] noted that 80% of all file creations have a lifetime of less than three minutes; all of these are missed.

Tables 1 and 2 summarizes the number of events within the portions of the traces analyzed in the paper. For the Sprite file system traces, four hours worth of events from the most heavily used server, which had over 75% of the Sprite cluster's file system events. Access privileges (i.e., read, write, or read/write access) were not considered for Sprite events in this section of the paper. Four million events were analyzed from the Auspex traces — this represents approximately 25 hours worth of trace data. All events from the INS and the UMBC traces were considered.

4.1 Visualizing the Traces

Figure 1 illustrates the INS traces across 6 orders of magnitude of time scales. Each plot within the figure represents file system event activity, enumerated as the number of events per time unit. Successive plots are refinements of the previous plots; the top plot in each column has a time unit of 32,768 seconds, the second of 4,096 seconds, and so on. The left column illustrates file system read or write events (summarized as XFER events), and the right column illustrates attribute read or write events (ATTR).

The most obvious feature of the XFER events is that they appear bursty at all time scales, although the burstiness appears to smooth out at coarser time granularities. For example, at the 500-second time scale, the peak-to-average ratio is about 60:1, but by the 6 month time scale, the ratio drops to about 5:1. For the ATTR events, a similar but even more pronounced effect can be observed. At fine time granularities, very bursty, unpredictable behaviour occurs. For coarse granularities (3-day and above), the traces become extremely smooth and predictable, although a small num-

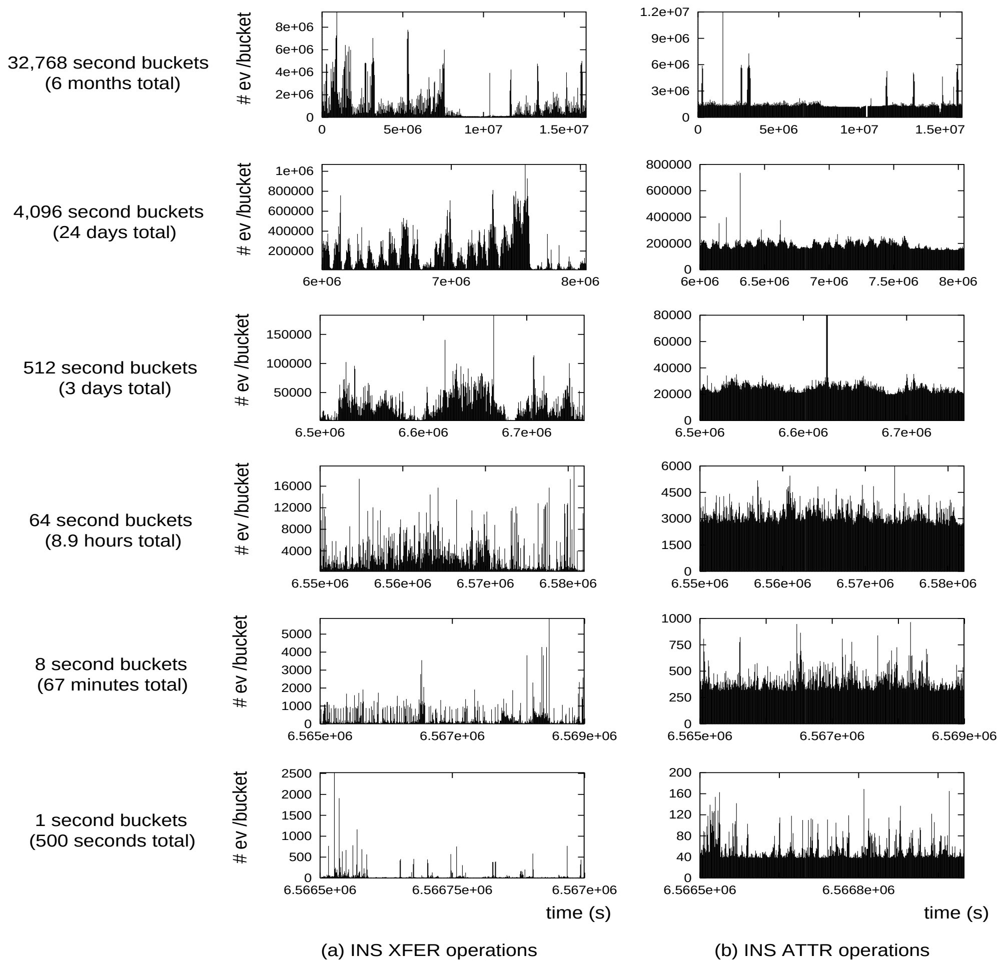


Figure 1: **Visualization of the traces:** INS file system events per unit time (or "bucket") are plotted for 6 increasingly refined time units, varying by a total factor of 32,768. Column (a) shows read and write operations, and column (b) shows file attribute read or modification events. The x-axis of all graphs is the number of seconds since the start of the trace that each bucket begins; the y-axis shows the number events that occurred during that bucket.

ber of large spikes can be seen to interrupt this smoothness.[3] A very strong diurnal cycle can also be seen in both event traces at the 3-day and 24-day scales.

Process models that are typically used while modeling systems events (such as those having Poisson or Markov modulated Poisson interarrival times) tend to look uniform at large time scales, primarily because of their finite variance and short-tailed distributions. In contrast, self-similar processes exhibit scale invariance; the processes will appear to be bursty at all time scales, and will not degrade to uniformity. Figure 1 contains characteristics of both process models; across short time scales (less than a day), the process appears bursty, while across larger time scales (more than a few days), the process smooths out to some degree.

[3]Closer examination of these large spikes at the coarse time scales revealed that they were caused by misbehaving or extremely poorly written programs. For example, a runaway program that was repeatedly reading a `.logout` file dominated all other file system activities for 2 days, causing a large spike to appear in figure 1.

5 Self-Similarity in File Systems (Short-term)

In this section, we demonstrate using rigorous statistical techniques the presence of self-similar behaviour in the two short-term file system traces (Sprite and Auspex NFS). The two techniques that we use are variance-time plots, and R/S analysis.

5.1 Variance-time plots

We can take advantage of equation 6 to more rigorously verify the self-similar nature of a process, and to estimate the value of the Hurst parameter H. Taking the logarithm of both sides of the equation results in the relation

$$\log\left(Var\left(X^{(m)}\right)\right) = c_1 - \beta \log(m) \tag{12}$$

for some constant c_1. Thus, plotting $\log\left(Var\left(X^{(m)}\right)\right)$ versus $\log(m)$ for many values of m of a self-similar process will result in a linear series of points with slope $-\beta$; this plot is known as a *variance-time plot*.

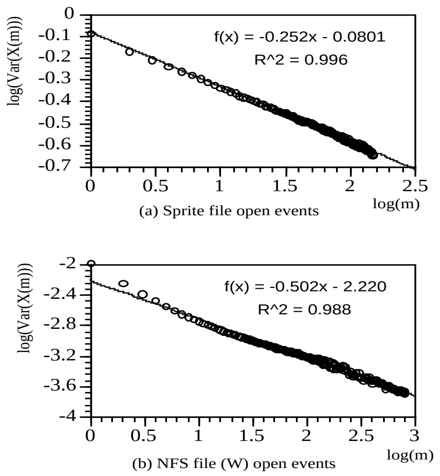


Figure 2: **Variance-time plots:** Variance plots for (a) Sprite file open events and (b) NFS file open (W) events are illustrated, and linear curve fits displayed.

Given a variance-time plot, an estimate of H can be obtained by calculating the slope β and using equation 5. Slopes between -2 and 0 correspond to Hurst parameters H between 0.5 and 1; if $0.5 < H < 1$, then the process is self-similar. Figure 2 illustrates the variance-time plot for Sprite file open events (2a) and NFS file open (W) events (2b). Both are extremely linear, and have Hurst parameters of 0.874 and 0.749, respectively. This verifies the self-similar nature of these processes.

Variance-time plots were generated for all of the events listed in table 1; the resulting Hurst parameter estimates are listed in table 3. In all cases except for open and close (RW) events within the NFS traces, all estimated Hurst parameters are well above 0.5, indicating that specific file system event types exhibit self-similarity across short time scales. The two cases for which the estimated Hurst parameter is low (NFS open and close events with RW privileges) correspond to events for which scant amounts of trace data were available (refer to table 1). These two estimated Hurst parameters are thus unreliable.

Somewhat surprisingly, the estimated values of H for the 1991 Sprite file system events seem to match their 1994 NFS counterparts. For example, the estimated value of 0.8734 for Sprite open events is quite close to the NFS open (R) value of 0.8413, although it is greater than the NFS open for write (W) and open read/write (RW) values of 0.7491 and 0.5286, respectively. Considering that the number of NFS open read (R) events dominate the NFS open (W) and open (RW) events, this is not unreasonable. However, as indicated by the confidence intervals on these H estimates, the differences between the NFS and Sprite values are statistically significant.

Unsurprisingly, the measured Hurst parameters for open events closely match the measured values for close events. Open events are quickly followed by close events; the open and close event processes should therefore be equally bursty and asymptotically self-similar.

5.2 R/S-Analysis and Pox plots

A second estimate of the Hurst parameter can be obtained through R/S analysis (originally presented in [13], and fully explained in [2]). Given a set of observations $(X_k : k = 1, 2, \ldots, N)$, that set is subdivided into K disjoint, contiguous subsets of length (N/K). The R/S statistic $R(t_i, n)/S(t_i, n)$ (equation 11) is then computed for the starting points t_i of the K disjoint subsets and for values of n satisfying the relationship $(t_i - 1) + n \leq N$.

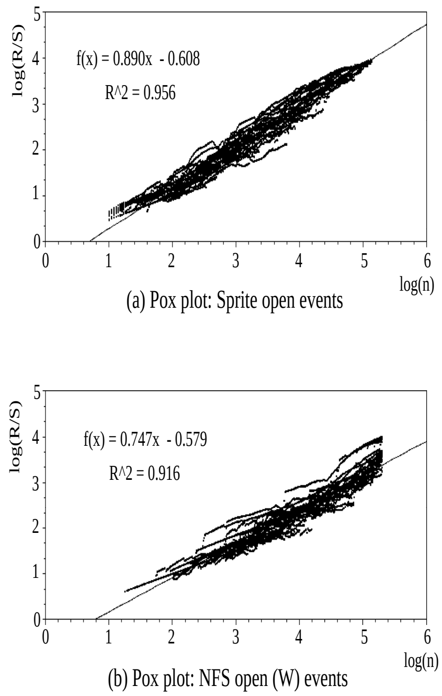


Figure 3: **Pox plots:** Pox plots for (a) Sprite file open events and (b) NFS file open (W) events are illustrated, and linear curve fits displayed.

From equation 11 can be derived:

$$\log\left(\frac{R(t_i, n)}{S(t_i, n)}\right) = c_2 + H\log(n). \tag{13}$$

Plotting $\log(R(t_i, n)/S(t_i, n))$ versus $\log(n)$ should therefore result in a roughly linear graph with a slope equal to the Hurst parameter H; such a graph is known as a *Pox plot*.

Figure 3 contains Pox plots for the same Sprite file open and NFS file open (W) events as were used to produce the variance-time plots of figure 2. A least-squares linear fit of the data produced estimated Hurst parameter values of 0.890 and 0.747, respectively. These estimates are extremely close to the previously estimated values of 0.874 and 0.749, which gives confidence to the variance and Pox plot analysis techniques as well as to the estimated values of H.

Estimates of H were generated through R/S analysis for all of the available file system events. Table 3 summarizes the results of this analysis. Once again, all estimated Hurst parameters are well above 0.5, confirming the presence of self-similarity in the file system traces. Furthermore, the

Event Type	Analysis Method	Sprite H	NFS H R	NFS H W	NFS H RW
Open	variance-time	0.8739 ± 0.0012	0.8686 ± 0.0055	0.7489 ± 0.0039	0.5220 ± 0.0016
	pox plot	0.8898 ± 0.0047	0.8521 ± 0.0052	0.7472 ± 0.0059	0.5696 ± 0.0039
Close	variance-time	0.8695 ± 0.0012	0.8753 ± 0.0070	0.7355 ± 0.0035	0.5268 ± 0.0023
	pox plot	0.8750 ± 0.0042	0.8464 ± 0.0050	0.7324 ± 0.0054	0.5675 ± 0.0039
Block Xfer	variance-time	n/a	0.8647 ± 0.0012	0.8681 ± 0.0050	n/a
	pox plot	n/a	0.8071 ± 0.0042	0.8424 ± 0.0077	n/a
Delete	variance-time	0.7773 ± 0.0045	n/a	0.8489 ± 0.0070	n/a
	pox plot	0.7463 ± 0.0040	n/a	0.7973 ± 0.0091	n/a
Attribute	variance-time	unused	0.8156 ± 0.0040	0.9899 ± 0.0007	n/a
	pox plot	unused	0.7578 ± 0.0048	0.7345 ± 0.0100	n/a
Directory	variance-time	unused	0.9832 ± 0.0005	n/a	0.7329 ± 0.0039
	pox plot	unused	0.9316 ± 0.0057	n/a	0.6659 ± 0.0054
Seeks	variance-time	0.8323 ± 0.0073	n/a		
	pox plot	0.7712 ± 0.0049	n/a		
All events	variance-time	0.9033 ± 0.0011	0.9688 ± 0.0008		
	pox plot	0.9098 ± 0.0044	0.8960 ± 0.0048		

Table 3: **Estimates of H:** This table summarizes the estimated values of the Hurst parameters H derived from variance-time and pox plots of the Sprite and NFS trace data.

correlation between all Pox plot generated H estimates and the previously presented variance-time plot generate H estimates is extremely high, although the estimates' confidence intervals do not overlap. With the exception of NFS attribute (W) events, all Pox plot and variance-time plot estimates were between 5-10% percent of each other.

The difference between the two measured H estimates for NFS attribute (W) events is startling, and cannot be easily explained. The uncharacteristically high value of the variance-time plot estimate (0.9934) suggests the possibility of an error in either the attribute (W) event traces, or perhaps is a result of the anomalies introduced during the NFS trace post-processing stage.

To summarize, we have used both variance-time and R/S analysis to convincingly show that short-term file system traffic exhibits self-similarity. The variance-time and R/S analysis produced consistent measures of the degree of burstiness of the two traces, and these measures appeared to be closely correlated across the Sprite and NFS traces.

6 Lack of Long-Term Self-Similarity in File Systems

Self-similarity by definition spans all time scales, so file system traffic over very long periods should still exhibit self-similar behaviour. However, as we demonstrate in this section, the self-similarity appears to break down at time scales in the neighbourhood of a day to a week. We saw some evidence of this in figure 1 when the INS trace became less bursty and more predictable at these time scales. We demonstrate this more convincingly here using variance-time analysis.

6.1 Variance-time Analysis of Long-Term Trace Data

Figure 4 shows the variance-time plots from three different traces. Figure 4(a) and (b) illustrate all and open file system events from the INS trace, respectively. Figure 4(c) shows all file system events from the UMBC trace. In all three plots, we can see that for fine-grained time scales, a linear relationship can be observed with a slope in the range $-1.0 < m < 0$, indicating self-similarity. However, at time scales approaching a day, this linear relationship breaks down, as evidenced by the "knees" in each curve.

This effect can also be observed in previously published work. In figure 4(d) we have replicated a previously published variance-time plot from [3] that illustrates self-similarity in world wide web traffic. This effect of non-linearity can be observed here; the plot just begins to fall off of the straight line as the dataset's time scales are exhausted. We hypothesize that if another order of magnitude of trace data had been analyzed, this effect would have been unmistakable.

This evidence, in combination with the visual confirmation of a smoothing out of the burstiness of the long-term traffic in figure 1, leads us to the conclusion that file system traffic is observably **not self-similar**—the burstiness of the traffic simply does not uniformly extend across all time scales. This is completely reasonable and intuitive: human behaviour dominates the traffic at the day through week level, as evidenced by the traces' unmistakably diurnal cycle. Across several weeks and months, there is some burstiness, but the burstiness is limited in scale and far less frequent than at short time scales. For such short time scales, however, the file system traffic is well-represented by a self-similar process, but for long time scales, self-similarity does not give a good representation of the traffic.

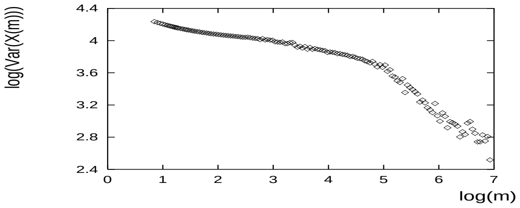
In [10], an analysis of a subset of heavy-tailed distributions[4] known as *Power-tail* distributions is presented. This work argues that the self-similarity identified in systems (including the Ethernet traffic from [9]) can be explained by an arrival process with such a power-tail distribution. Power-tail distributions do obey the central limit theorem, but only for extremely large aggregation values. [10] argues that Leland et al. would have observed their traffic instability to smooth out if only they had increased the time scale of their analysis by another 2 orders of magnitude to a total of 7.

In comparison, our INS traces ostensibly span a total of more than 6 orders of magnitude. The INS traces are unique in that they afforded us the possibility of observing such a wide time range of trace data that it resulted us in observing the cessation of the self-similar behaviour.

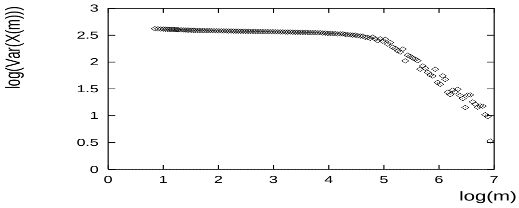
7 ON/OFF Sources

We have not yet attempted to explain the underlying cause of the short term self-similar behaviour of file-system traffic. Willinger *et al.*[23] proposed a physical explanation of observed self-similarity in Ethernet LAN traffic, based on the-

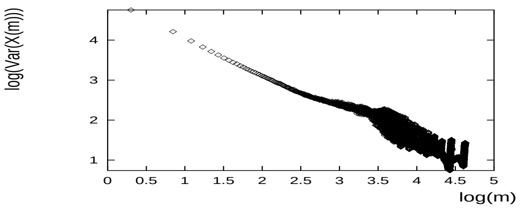
[4] A heavy-tailed distribution is typically one which exhibits infinite variance. An example of a heavy-tailed distribution is the Pareto distribution, whose general form is $P(x) = \beta a^{\beta} x^{-\beta-1}$ with $a, \beta \geq 0$ and $x \geq a$. The Pareto distribution has infinite variance for values of $\beta < 1$.



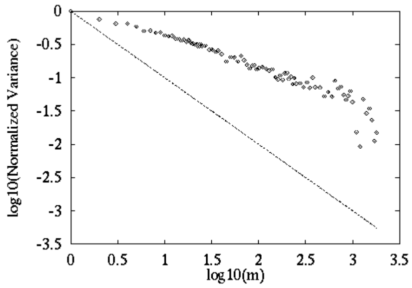
(a) INS trace, all events



(b) INS trace, open events



(c) UMBC trace, all events



(d) Boston University WWW trace

Figure 4: **Variance-time plots:** these plots clearly demonstrate a substantial change in traffic characteristics at a time scale between 10^4 and 10^5 seconds per bucket, i.e. at the day to week long time scale. The first two figures (a) and (b) are of INS trace events, and the second is of the UMBC trace events. (d) is a reproduced figure from a published paper on self-similarity in World Wide Web traffic [3].

ory developed initially by Mandelbrot[11] and then Taqqu and Levy[20].

The theory states that the aggregation of many ON/OFF sources, each exhibiting a characteristic known as the *Noah effect*, results in self-similar total traffic. An individual source is classified as being ON/OFF if its traffic appears to contain highly variably lengthed ON and OFF periods; an ON period contains much activity, while an OFF period has a complete lack of activity. The Noah effect refers to the high variability of the ON and OFF periods. If the distribution of ON and OFF period lengths from individual sources is heavy-tailed, then the aggregate traffic exhibits the Noah effect, and can be shown to exhibit self-similarity.

The theory presented in [20] makes the simplifying assumption that events within an ON period are evenly distributed. The ON/OFF source model is thus similar to packet-train models often used to model network sources, but with the exception that packet interarrival times must have a heavy-tailed distribution.

7.1 ON/OFF Behaviour in the Short-term Traces

An examination of the trace events from individual source hosts within the Sprite cluster and NFS installation should identify whether or not the self-similarity of the file system traces can be explained using this ON/OFF model. Figure 5 presents two textured dot strip plots obtained from the Sprite and NFS traces. A textured dot strip plot (proposed in [21]) is a two-dimensional representation of a one-dimensional time-series. Each vertical column in a plot corresponds to one time unit; the displacement of dots (representing events) within that column represents the fractional position of the event in that time unit.

In order to better depict the ON/OFF behaviour of the sources, the subset of the Sprite traces that was analyzed was extended from a four hour subset to an approximately 41 hour subset; access privilege information was also extracted from the Sprite traces. Figure 5a illustrates the textured strip dot plot for Sprite open (W) events originating from host workstation ID 42 (4168 such events were extract from the trace.) Similarly, figure 5b illustrates NFS block reads from host workstation ID 3068, for which 23025 events were extracted. These two hosts were chosen because they were both quite active throughout the period of analysis, and their high activity resulted in visually striking ON/OFF periods within the dot plots. It should be noted that even relatively lightly active hosts could be seen to exhibit this ON/OFF behaviour, although the behaviour was not as pronounced in their textured dot plots.

The ON/OFF behaviour of these two sources is unmistakable. It is also clear that the ON periods for NFS block read events are much sharper and more dense than for Sprite file open (W) events. This difference is easily explainable. First and foremost, files are known to be read far more frequently than they are written[1]; from table 1 we see that approximately 13.4% of NFS files are opened with write or RW privileges, while 86.6% of files are opened with read-only privilege. Similarly, [1] reported that 88% of files within the Sprite traces were opened with read-only privilege. Secondly, many file read and write events occur in between a given file open and close pair. The total amount of read and write events therefore greatly outnumbers the amount of open or close events.

7.1.1 The Analysis of ON and OFF periods

In order to verify the presence of the Noah effect, the ON and OFF periods for these sources first need to be identified. To do so, we use a method similar to that described in [23]. The source's trace is scanned linearly; given an event from the trace, we assume that subsequent events belong to the same ON period if they occur within some threshold amount

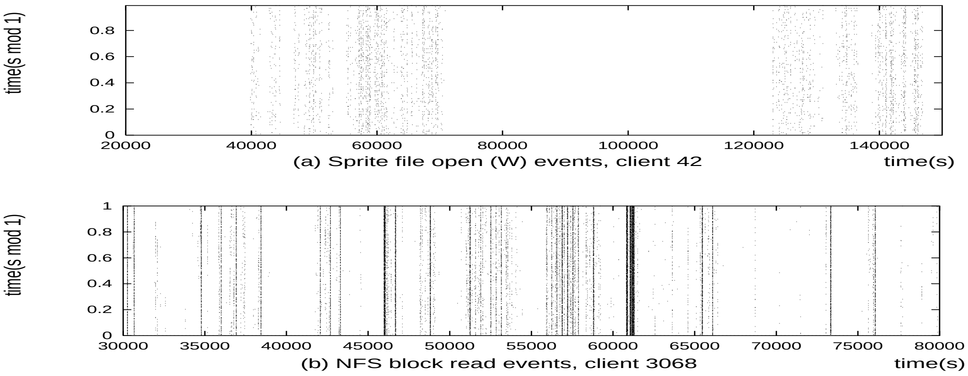


Figure 5: **Textured dot strip plots:** Textured dot strip plots with time units of 1 second for the (a) Sprite file open (W) events of host 42 and (b) NFS block read events of host 3068 are illustrated.

of time, otherwise we mark the interval to the next event as an OFF period. The ON/OFF period size distributions were calculated for the Sprite file open (W) events, using a threshold value of 60 seconds.[5] The resulting number of ON and OFF periods was 182 and 181, respectively.

If both the ON and OFF period length distributions are heavy-tailed, i.e. they satisfy

$$P(U > u) \sim cu^{-\alpha} \text{ with } u \to \infty, 1 < \alpha < 2, \tag{14}$$

for period length U, and if the activity within an ON-period is uniform, then the aggregation of many such sources results in a self-similar process with Hurst parameter $H = \frac{3-\alpha}{2}$. [23]

Once the ON and OFF periods from a given source have been identified, we can verify the presence of the Noah effect using complementary cumulative distribution plots, or "qq-plots."[8] The idea is simple: if we assume that the period length distribution under analysis obeys equation 14, then by taking the log of both the sides of equation, we obtain

$$\log P(U > u) \sim \log(c) - \alpha \log(u) \text{ as } u \to \infty. \tag{15}$$

Plotting the measured complementary cumulative period length distribution $P(U > u)$ versus period length u in a log-log plot should thus yield a straight line with slope $-\alpha$, for large enough values of u. We performed this analysis, and did indeed observe a linear relationship, with a slope of approximately -1.16. We also used a second method for identifying the Noah effect known as a *Hill plot.* We omit details of this analysis, but merely state that its results confirm those of the qq-plot. Interested readers are referred to [23] for a treatment of Hill plots.

Both the Hill and qq-plot analysis confirm that the observed ON and OFF periods lengths are Pareto distributed. This is consistent, since we have already shown that the aggregate traffic of all observed ON/OFF sources from the short-term traces is self-similar in nature.

7.2 Synthesizing Self-Similar Trace Data

The ON/OFF source behaviour for individual clients in the short-term traces suggests a simple method for synthesizing a stream of self-similar events. We can model an individual client's activity as a set of ON/OFF periods, i.e. a packet train with heavy-tailed train lengths and inter-train arrival times. We can then aggregate many such synthesized clients, and the resulting traffic should exhibit the desired self-similar behaviour.

In section 7.1, we demonstrated that the file open (W) event series generated by a client exhibits ON-OFF behaviour. Our simple model therefore generates ON and OFF periods whose lengths are Pareto distributed and populates the ON periods by generating a succession of file open events within each of these periods. To do this correctly, we require knowledge of the distribution for inter-arrival times between file-open events. Having generated an open event, we then generate subsequent write events and finally a close event.

In order to test our simple packet train model, we extracted a number of distributions from the Sprite and NFS file system traces to use as inputs to the model. The parameter α extracted from the Hill plot in section 7.1.1 was used to create ON/OFF periods. In order to populate the ON periods with open events, the distribution of file open (W) inter-arrival time was measured and used.

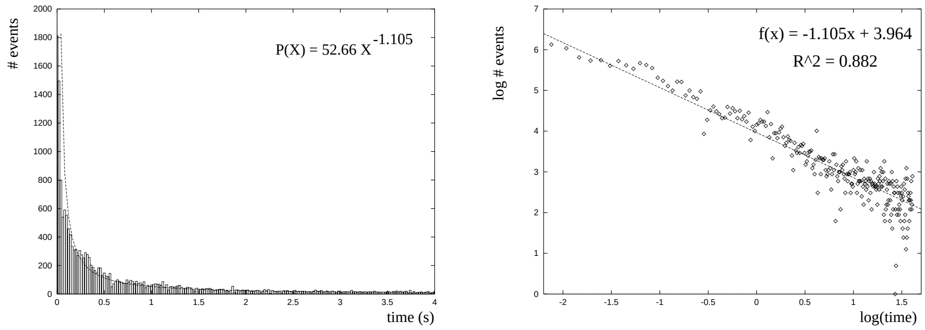
Figure 6a shows the measured distribution of file open (W) events from the Sprite traces.[6] The illustrated distribution was truncated at 4 seconds for the sake of clarity, but in reality it extended as far out as 900 seconds. Clearly this distribution is heavy-tailed (which is consistent, since we previously demonstrated that Sprite open events are self-similar). In order to model this distribution, we fit the data against the general form of a Pareto distribution:

$$P(x) = c_3 \, x^{-\alpha}. \tag{16}$$

Taking the logarithm of both sides of equation 16 results in the linear relation $\log(P(x)) = \log(c_3) - \alpha \log(x)$. Thus, a log-log plot of the measured distribution should result in a linear function with slope $-\alpha$ and intercept $\log(c_3)$. Figure 6b illustrates the log-log plot of distribution 6a. The least-squares fit of this plot resulted in an estimated α value of 1.105 and c_3 value of 52.66. These values were then substituted into equation 16, and the resulting distribution overlayed on top of figure 6a. The strong resemblance between this generated distribution and the observed data indicates that the choice of a Pareto distribution as a model for file

[5] The threshold value of 60 is somewhat arbitrary. We observed a large range in potential threshold values that gave very similar results; the value 60 gave representative results.

[6] We chose to measure distributions from the Sprite file traces in order to avoid the post-processing anomalies present in the NFS traces.



(a) The distribution of Sprite file open (W) interarrival times

(b) Log-log plot of distribution a

Figure 6: Sprite file open (W) inter-arrival times

open inter-arrival times is reasonable. Note that for the purposes of modeling, only the $\alpha = 1.105$ value is important, since c_3 is only an indicator of the number of events in our measured distribution, and is replaced by a normalization constant in the model.

Given the value α, we can now generate file open events within ON periods. To generate the file close events to match the open events, we need the distribution of times between file open events and their corresponding closes. A similar analysis revealed that this distribution (for the Sprite traces) was also well represented by a Pareto distribution with $\alpha = 1.131$. The distribution of file sizes (measured at the time of close (W) events) was also seen to be heavy-tailed. Analysis of this distribution resulted in an estimated $\alpha = 1.481$.

7.2.1 Model Results

To test the validity of our model, we synthesized file system events for 20 clients and aggregated them to obtain the overall traffic for the server. The textured plot for all events generated by a representative individual client (number 15) is shown in figure 7. The plot looks remarkably similar to the ones encountered by us during our studies. To verify that the OFF period lengths for this source indeed follow the Pareto distribution, we generated a qq-plot and Hill plot for a synthesized client. The estimate of α that we measured using both techniques was approximately 1.25, which is the same value that had been used as a parameter to our model.

To test the behaviour of the aggregate traffic, we performed R/S analysis and variance-time analysis. The value for the Hurst parameter calculated from both is 0.615 and 0.697 respectively — this shows that the aggregate synthesized traffic is self-similar. The strength of our model therefore lies in being able to exhibit the same bulk properties as those of the traces input to it.

7.3 ON/OFF Summary

We have shown that:

- file system clients from both the NFS and Sprite file system traces show ON/OFF behaviour.
- the distribution of ON/OFF period lengths in both traces is heavy-tailed, and therefore the traces show the Noah effect.
- it is possible to use this ON/OFF behaviour as the motivation for a simple packet train model that can be used to synthesize traffic that exhibits self-similar behaviour.

8 Conclusions

Previous studies have demonstrated that byte or packet level network traffic is self-similar in nature. In this paper, we have shown that high-level file system events also exhibit this self-similar behaviour, but only across time scales of less than a day. The total file system traffic and individual classes of file system events (such as file opens, closes, writes, etc.) seen by the file server were both observed to exhibit this self-similar behaviour. This "short-term self-similarity" was observed in four file system traces (of a 1991 Sprite distributed system cluster, a 1994 Auspex NFS installation, a 1996/1997 instructional UNIX cluster, and a 1997 NFS installation) with significantly different characteristics. The load, typical application usage, tracing technique, and dates of tracing all differed between the three traces, but the measured self-similarity parameters were observed to be relatively consistent across all traces.

Two of the traces were then analyzed across a much wider set of time scales, and the self-similar behaviour ceased for time scales of days, weeks, and months. Although burstiness did persist, the bursts began to smooth out. This smoothing out had the effect of showing up as very pronounced non-linearities in the variance-time plots of the long-term traces.

A proposed causative factor of short-term self-similarity, namely ON/OFF source behaviour, was also observed for the Sprite and Auspex traces. Individual sources of file system activity (i.e. workstations) were seen to have the hypothesized behaviour, and an analysis of the ON/OFF period lengths demonstrated them to exhibit the Noah effect required for aggregate self-similar traffic. Both the ON/OFF behaviour of individual sources and the resulting short-term self-similar aggregate file system traffic are intrinsic to file systems, and must be considered during the design and simulation of file systems. We demonstrated that this ON/OFF behaviour could be used to synthesize a stream of file system events that show the desired short-term self-similarity.

In conclusion, the four file system traces that we analyzed had significant differences: the file system being traced, the method of gathering the traces, the chronological time that the traces were gathered, the information gathered within the traces, and the user and system environments all dif-

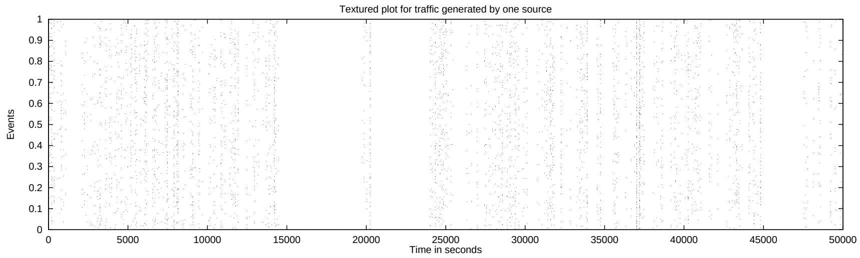


Figure 7: **Textured dot strip plot:** All synthesized file system events generated for client 15 over 50,000 seconds.

fered. Despite these differences, we found the same short term self-similar behaviour in all three traces. However, since we found that this self-similar behaviour is limited to short time scales, we must conclude that file system traffic is, strictly speaking, not self-similar in nature.

Acknowledgments

We would like to thank Mike Dahlin for giving us access to and help with the Sprite and Auspex file system traces. We also gratefully acknowledge the suggestion to investigate ON/OFF behaviour provided by Venkat Padmanabhan, and the helpful comments of Jeanna Neefe and Eric Anderson.

References

[1] Baker, M. G., Hartman, J. H., Kupfer, M. D., Shirriff, K. W., and Ousterhout, J. K. Measurements of a Distributed File System. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles* (1991).

[2] Beran, J., Sherman, R., Taqqu, M. S., and Willinger, W. Long-range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transactions on Communications 43* (Mar. 1995).

[3] Crovella, M. E., and Bestavros, A. Explaining world wide web traffic self-similarity. Tech. Rep. TR-95-015, Computer Science Department, Boston University, Oct 1995.

[4] Dahlin, M. D., Mather, C. J., Wang, R. Y., Anderson, T. E., and Patterson, D. A. A quantitative analysis of cache policies for scalable network file systems. In *Proceedings of the SIGMETRICS '94 Annual Conference on Measurement and Modeling of Computer Systems* (Nashville, Tennessee, May 1994).

[5] Ferrari, D., and Verma, D. C. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications 8*, 3 (1990).

[6] Georganas, N. D. Self-similar ("fractal") traffic in atm networks. In *Proceedings of the 2nd International Workshop on Advanced Teleservices and High-Speed Communications Architectures (IWACA '94)* (Heidelberg, Germany, Sept. 1994), pp. 1–7.

[7] Hartman, J. H. *Using the Sprite File System Traces.* Computer Science Division, EECS Department, University of California at Berkeley, May 1993.

[8] Kratz, M. F., and Resnick, S. I. The qq-estimator and heavy tails. Preprint, 1995.

[9] Leland, W. E., Taqqu, M. S., Willinger, W., and Wilson, D. V. On the Self-Similar Nature of Ethernet Traffic (extended version). *IEEE/ACM Transactions on Networking 2* (Feb. 1994).

[10] Lipsky, L. Modelling telecommunications systems that have power-tail (bursty, chaotic, heavy-tail, self-similar, etc.) traffic. In *Proceedings of the 22nd International Conference on Technology Management and Performance Evaluation of Enterprise-Wide Information Systems (CMG96)* (San Diego, CA, USA, December 1996).

[11] Mandelbrot, B. Self-similar Error Clusters in Communication Systems and the Concept of Conditional Stationarity. *IEEE Transactions on Communication Technology COM-13* (1965).

[12] Mandelbrot, B. *The Fractal Nature of Geometry.* W. H. Freeman and Company, New York, NY, USA, 1983.

[13] Mandelbrot, B. B., and Wallis, J. R. Computer experiments with fractional gaussian noises. *Water Resources Research 5* (1969).

[14] Ousterhout, J. K., Costa, H. D., Harrison, D., Kunze, J., Kupfer, M., and Thompson, J. A Trace-driven Analysis of the UNIX 4.2 BSD File System. *Operating System Review 19*, 4 (1985), 15–24.

[15] Paxson, V. Fast approximation of self-similar network traffic. Tech. rep., Lawrence Berkeley Laboratory and EECS Division, University of California, Berkeley, 1 Cyclotron Road, Berkeley, CA 94720, Apr. 1995.

[16] Paxson, V., and Floyd, S. Wide-area Traffic: the Failure of Poisson Modeling. In *ACM SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications* (London, UK, Aug. 1994).

[17] Roselli, D. Characteristics of file system workloads. Tech. rep., The University of California at Berkeley, 1998.

[18] Rosenblum, M., and Ousterhout, J. K. The Design and Implementation of a Log-Structured File System. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles* (1991).

[19] Smith, K., and Seltzer, M. File layout and file system performance. Tech. Rep. TR-35-94, Harvard University, 1994.

[20] Taqqu, M., and Levy, J. Using renewal processes to generate long-range dependence and high variability. In *Dependence in Probability and Statistics* (Boston, MA, 1986), E. Eberlein and M. Taqqu, Eds., pp. 73–89.

[21] Tukey, J., and Tukey, P. Strips displaying empirical distributions: I. textured dot strips. In *Bellcore Technical Memorandum* (1990).

[22] Wilkes, J., Golding, R., Staelin, C., and Sullivan, T. The hp autoraid hierarchical storage system technology. In *Proceedings of the 15th Symposium on Operating System Principles* (Monterey, CA, USA, Nov 1995).

[23] Willinger, W., Taqqu, M. S., Sherman, R., and Wilson, D. V. Self-similarity through high-variability: Statistical analysis of ethernet lan traffic at the source level. In *ACM SIGCOMM '95 Conference on Communications Architectures, Protocols and Applications* (Cambridge, MA, USA, 1995).