The Relevance of Long-Range Dependence in Disk Traffic and Implications for Trace Synthesis

Bo Hong Tara M. Madhyastha

{hongbo,tara}@soe.ucsc.edu
Department of Computer Engineering
University of California Santa Cruz
1156 High Street
Santa Cruz, CA 95064

Abstract

Accurate disk workloads are crucial for storage systems design, but I/O traces are difficult to obtain, unwieldy to work with, and unparameterizable. Unfortunately, I/O traces are extremely bursty and difficult to characterize. Although good models of I/O workloads would be extremely useful, traces cannot accurately be modeled using exponential or Poisson arrival times. Much experimental evidence shows that I/O traces are self-similar, which researchers have hoped might help to model bursty traces. In this paper, we show that self-similarity at large time scales does not significantly affect disk behavior with respect to response times. This allows us to generate synthetic arrival patterns at relatively small time scales, improving the accuracy of trace generation. The relative error of our method, with input parameters suitable for the workload, ranges from approximately 8% to 12%.

1 Introduction

Performance analysis and architecture of storage systems depends heavily upon traces and simulation. Unfortunately, I/O traces are difficult to obtain, extremely large and unwieldy, and cannot be parameterized. Benchmarks and models are simpler and more workable alternatives to traces, but are less realistic than the actual workloads.

Ideally, one would like to monitor any disk workload and model it accurately (with respect to some important performance metrics) with some small number of parameters. This vision is far from reality; however, this paper identifies parameters that can capture request interarrival burstiness.

We consider an I/O trace that consists of a set of timestamped values each containing a disk offset, a read/write flag, and a length. This low-level description accommodates a primitive application-level or SCSI I/O interface. We wish to model these streams accurately enough so that accesses synthesized from the model cause a storage hierarchy to behave "similarly" to the real trace. For a single disk drive, similar behavior is measured by checking that the distributions of queue lengths and response times resemble those created by the original workload.

We show that self-similarity at large time scales does not significantly affect disk behavior. This allows us to generate synthetic interarrival patterns at relatively small time scales, improving the accuracy of trace generation.

The paper is organized as follows. We describe related work in §2. We introduce self-similarity as a way to approximate long-range dependence and show that long-range dependence has little effect upon disk

response times in §3. Binomial multifractals can generate bursty traffic; §4 describes this model. In §5 we describe a novel I/O request synthesis technique using multifractal models. We evaluate our method in §6 and conclude with directions for future work in §7.

2 Related Work

Generating realistic disk traces is a difficult and unsolved problem [5]. I/O traces are extremely bursty and difficult to characterize. They cannot accurately be modeled using exponential or Poisson arrival times, but there is strong evidence that the distribution of disk I/O, file, network, and Web traffic is self-similar [10, 4, 7, 6].

Several researchers have used self-similarity to model bursty traces, particularly network traces. Chen *et al* [2] examined ATM variable bit rate traffic and found that the higher the Hurst coefficient, a measure of self-similarity, the burstier the traffic. However, multifractal models, or generalizations of self-similar traffic models, have been shown to model some kinds of traffic more effectively than self-similar models [3]. Self-similarity alone does not necessarily capture burstiness, which has a significant effect on disk performance. To address this problem, Wang *et al* [17] proposed to use binomial multifractals to model the bursty disk traffic. The model is parsimonious, depending only on a single parameter, the bias p, which can be estimated from the traces.

Grossglauser and Bolot [8] demonstrated that it was not useful to model long-range dependence in network traffic at timescales disproportionate to the performance metrics under observation. Neidhardt and Wang [11] showed that queuing behavior depends not only on the Hurst coefficient, but a combination of system parameters. Our approach is to investigate whether this is true for I/O traffic, and whether this fact is useful for improving multifractal synthesis techniques.

3 Relevance of Long-Range Dependence in Disk Traffic

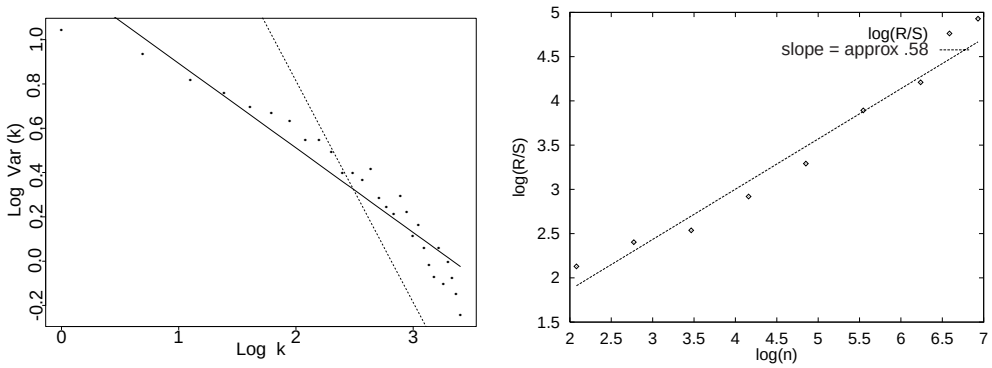
I/O workloads have a structure that researchers have proposed might help to model them called *self-similarity*. Informally, in this context, to say a time series is self-similar implies that it looks qualitatively the same at different time scales. Self-similar traffic also has the property of *long-range dependence*: the data set exhibits a slow decay in its autocorrelation function. This correlation structure is significant because self-similar traffic may be more bursty than that generated by other sources. However, we show here that long-range dependence, as measured by the Hurst coefficient, has little effect upon disk response times.

3.1 Self-Similarity

A more rigorous definition of self-similarity from [1] is as follows: Let Y_t be a stochastic process with continuous time parameter t. Y_t is called self-similar with self-similarity parameter H, if for any positive stretching factor c, the rescaled process with time scale ct, $c^{-H}Y_{ct}$, is equal in distribution to the original process Y_t.

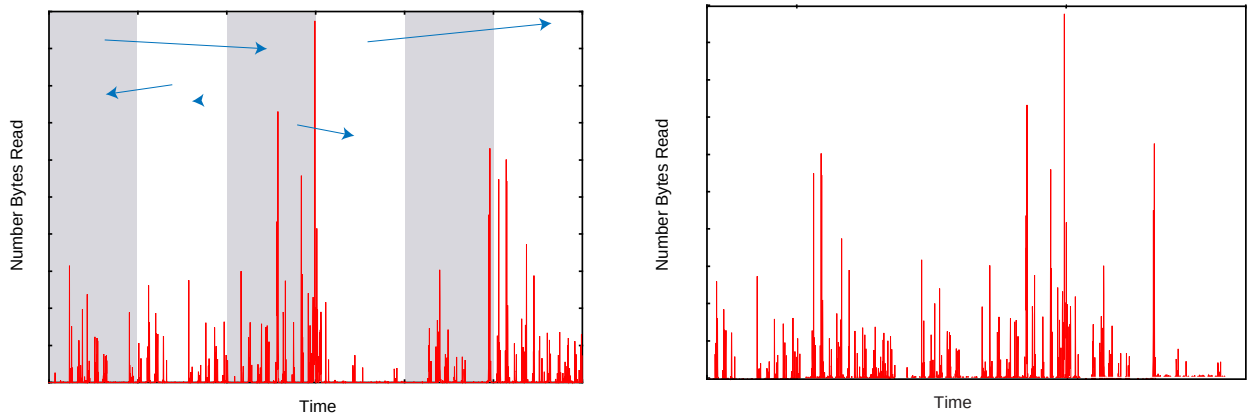
The parameter H is also known as the Hurst coefficient, and a value of H between $\frac{1}{2}$ and 1 indicates the degree of self-similarity. There are several exploratory analytic tools that are used to estimate H; two such methods are applied to a small UNIX workstation disk trace [13] in Figures 1a and 1b.

The first method, shown in Figure 1a, is a variance plot. We plot the logarithm of the variance of an aggregated (averaged) series against the logarithm of the aggregation level. The slope of this plot should be equal to $H - 1$. The second method, shown in Figure 1b, is the R/S plot. This method plots (in logscale) the R/S statistic, or the *rescaled adjusted range* against the log n. The rescaled adjusted range is the data range



(a) (b)

Figure 1: Estimating the Hurst parameter.



(a) (b)

Figure 2: Shuffling traces removes long-range dependence.

normalized by the standard deviation. For the precise definition of how to calculate this statistic, see [1, 16]. If there is long-range dependence in the process, the slope of the curve generated by this plot provides an estimate of H; if not, $\log R/S$ should be randomly scattered around a straight line with slope 0.5 [1].

3.2 Long-Range Dependence in Disk Traffic

Our hypothesis is that previous events cannot affect disk behavior beyond a certain threshold, determined by system parameters, so modeling long range dependence at larger timescales is unnecessary. To test this hypothesis, we study how these metrics change as we gradually destroy long-range dependence in the traces by shuffling increasingly smaller intervals. This approach is identical to the experimental approach taken by [8], and is illustrated in Figure 2. Figure 2a shows a trace that has been divided into six intervals. These intervals are then randomly rearranged to create a new trace (Figure 2b). Within each interval, the temporal relationships are preserved, but the new trace has no long-range dependence beyond the width of the interval.

Our selected workloads, described in more detail in [13], are the cello news disk trace (HP2204A) and the snake usr2 disk trace (HP97560) gathered between 05/30/92 and 06/06/92. The average I/O loads on the disks on these systems are small: approximately three requests for cello news disk and one request for snake

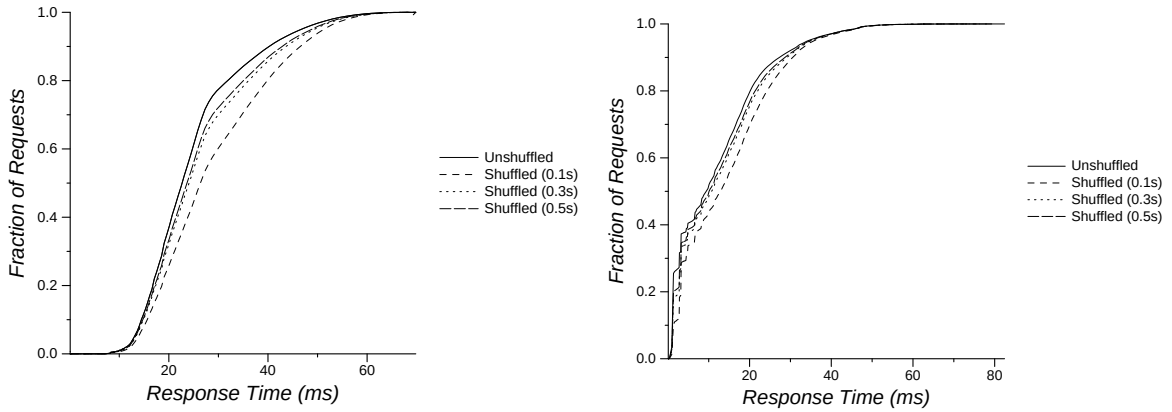


Figure 3: Response times for traces shuffled using small intervals for (a) cello and (b) snake.

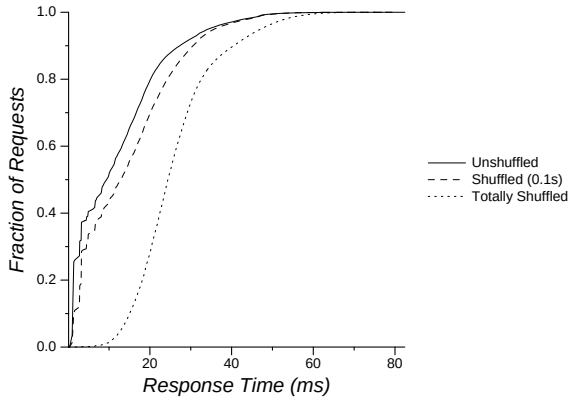


Figure 4: In the limit, all temporal locality is lost when we shuffle snake traces.

usr2 disk per second. However, the maximum queue lengths can be very large: over 1000 requests on the cello news disk and over 60 requests on the snake usr2 disk. In general snake traces are more bursty than cello, and the logical sequentiality (percentage of requests that are at adjacent disk addresses or addresses spaced by the file system interleave factor) of cello and snake is 2% and 29%, respectively.

We examine the numerical metric of disk performance used in [14] to validate disk models: the root mean squared (RMS) horizontal distance between the cumulative distribution functions (CDF) of I/O response times. The distributions of queue lengths of traces shuffled at intervals > 1 second are similar to those of the real traces [9], and are not presented separately.

We vary the shuffle interval length from 10 seconds to 0.1 seconds and use both the shuffled and unshuffled traces to drive the Pantheon [18] disk simulator.

Figure 3 shows the CDF of response time for small intervals. We can see that shuffling traces using intervals smaller than 0.5 seconds results in extremely skewed distributions of response time. In the limit, we destroy all temporal locality by randomizing all events, and obtain a curve as shown in Figure 4. In contrast, although at an interval size of 1 second there is virtually no self-similarity left in the trace, the relative error is very small, as shown in Figure 5.

Figures 6a and 6b show the Hurst coefficient, estimated using the R/S method, and the relative error for

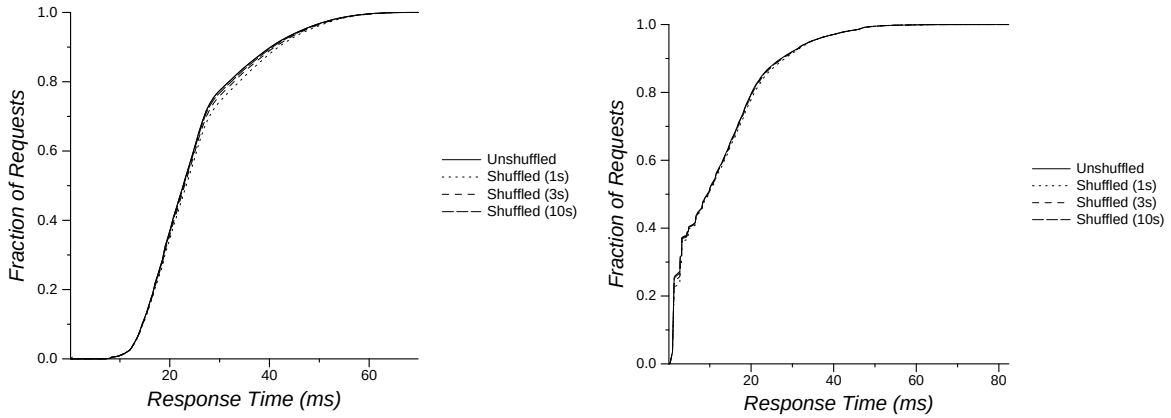


Figure 5: Response times for traces shuffled using intervals of length 1 second and above for (a) cello and (b) snake.

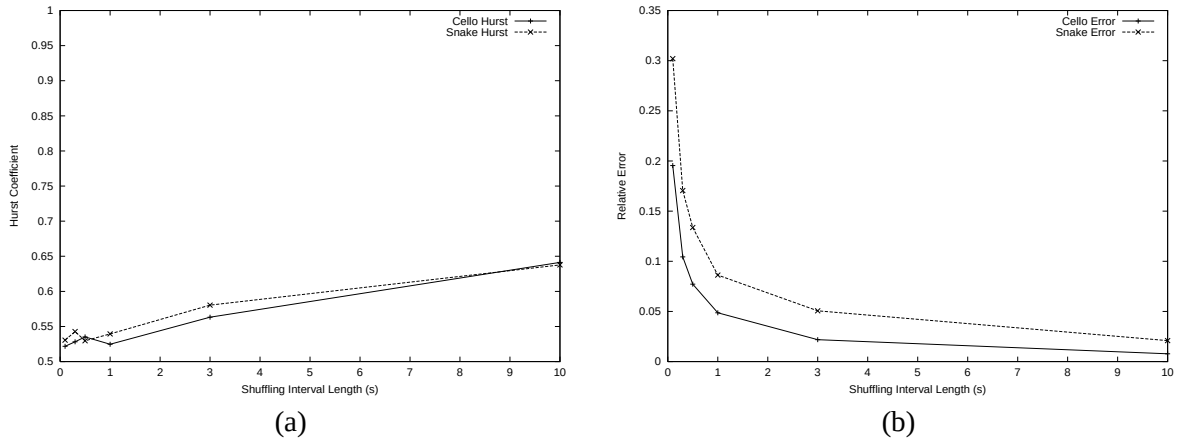


Figure 6: Effect of shuffling on (a) Hurst coefficient and (b) relative error.

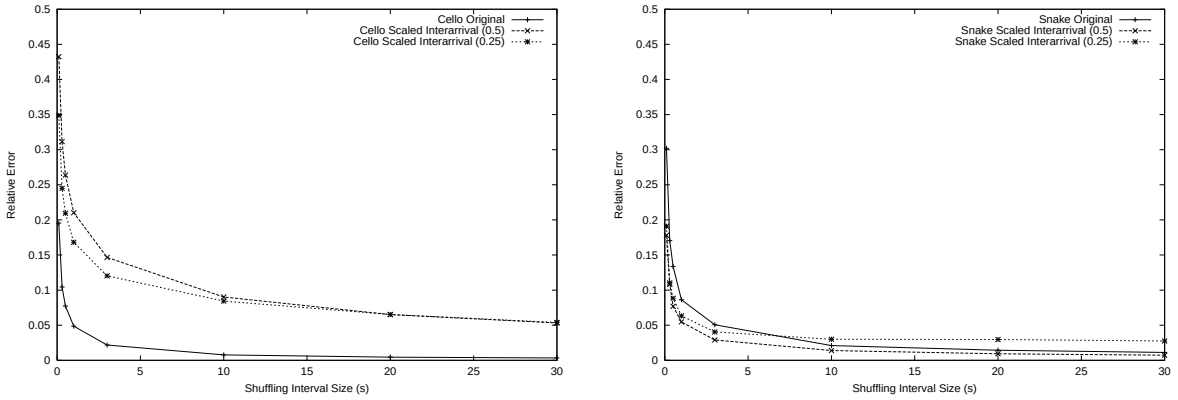
the shuffled traces. The Hurst coefficient is a measure of long-range dependence; as intuition dictates, the smaller the time interval, the fewer long-range correlations are preserved and the lower the Hurst coefficient. For one day, $H = .79$ for snake and $H = .89$ for cello.

Despite the lack of long-range dependence, particularly indicated by the fluctuation of the Hurst coefficient at small intervals (< 1 second), the relative error for the shuffled traces is relatively small; at 1 second it is approximately 5% for cello and 9% for snake.

3.3 Choosing an Interval

We see from Figure 6 that the relative error for the two traces at various interval lengths is different, and is larger for snake than for cello. To better understand how to select an appropriate interval length to bound the error for each trace, we studied the relative error as a function of burstiness. To artificially increase the "burstiness" of the trace, creating more disk request queuing, we scale down the interarrival time by a factor of 2 or 4.

Figures 7a–7b show the the effect of scaling on relative error for cello and snake, respectively. In



(a) (b)

Figure 7: Effect of shuffling on relative error for scaled traces of (a) cello and (b) snake.

general, as the shuffling interval length increases, the relative error decreases. However, we see that the interval length necessary to maintain the same relative error does not automatically increase as we scale down the interarrival time, and that the curves are quite different for the two machines.

Cello has fewer long "gaps" between activity than snake (see Figure 10), so shuffling has less effect on this distribution. Cello requests are less sequential than snake, so shuffling does not perturb the spatial locality for cello as much as for snake. Thus, the error caused by shuffling the original trace is lower for cello than for snake.

When the interarrival times are shortened, queue lengths increase. For cello, this improves average seek time because the scheduler can optimize requests. Shuffling changes this queuing behavior and causes higher errors than in the original trace. For snake, this queuing effect is not as important because snake has a disk cache and more sequentiality than cello. Thus, errors for the shuffled scaled traces are actually lower than that of the shuffled original trace at small intervals, and increase with larger shuffling intervals.

3.4 Modern Traces

The traces described here are from 1992. A re-configured cello was re-traced in 1999, but we have not yet been able to repeat our experiments on those traces. However, to generalize our results on long-range dependence to modern traces, we studied the characteristics of the new cello news disk traces (obtained from a Seagate ST19171W disk) for one week (09/09/1999 to 09/15/1999). The I/O load has increased to about 16 requests per second but the maximum queue length is in the same range as cello in 1992, from 700 to 1300. The logical sequentiality of cello (1999) is less than 1%. The Hurst coefficient is .89, similar to the 1992 cello traces. Experiments on 1992 cello traces to compress the interarrival times (Figure 7) approximate the modern traces with respect to Hurst coefficient and mean interarrival times. We believe that the shuffling interval required to minimize error for modern traces may be slightly longer, but otherwise the behavior is the same.

3.5 Summary

We can conclude from our experiments that self-similarity at large time scales does not significantly affect disk behavior with respect to response times. For purposes of performance evaluation, we need only consider I/O activity at timescales related to the system we are evaluating. For measuring disk response time and queuing behavior, an appropriate interval length was estimated empirically to be between 3 and 10 seconds

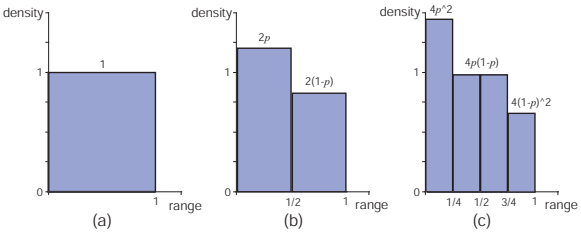


Figure 8: Recursive process in binomial measure generation. Start from (a) with a uniform probability measure, divide the mass with probability p in (b), divide again in (c).

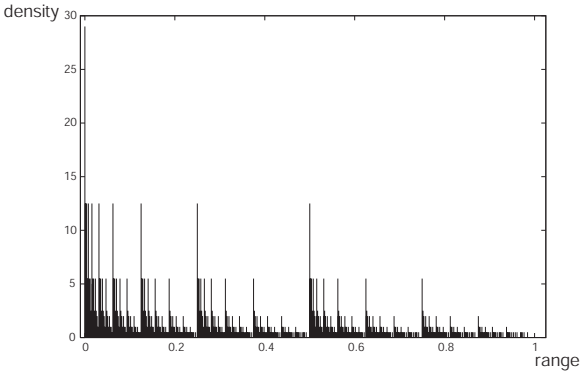


Figure 9: Probability density function for a sample binomial multifractal with bias .7.

for both the HP97560 and HP2204A disks under two different workloads.

4 Binomial Multifractals

Self-similarity is a measure of fractal-like scaling behavior over multiple time scales, characterized by the single Hurst parameter. In contrast, multifractals are a generalization of monofractal self-similar processes that allow for time-dependent scaling laws, and are based on multiplicative schemes. They have a bursty appearance similar to that of real I/O traffic. We introduce binomial multifractals, for the purpose of modeling I/O traffic, below. A rigorous introduction to binomial measures and multifractals can be found in [12].

4.1 Property of Self-Similarity

We can define a binomial measure on the unit interval in a recursive construction. Figure 8 shows the first two stages of the construction, which starts with the uniform probability measure μ_0 on the unit interval I = [0, 1] with mass 1 (Figure 8a). At the first stage (Figure 8b), I is split into two equal-length subintervals I_0 = [0, 1/2] and I_1 = [1/2, 1] and the masses $m_0 = p$ $(p > 1/2)$ and $m_1 = 1 - m_0 = 1 - p$ are spread uniformly between them. The density on I_0 and I_1 is $2p$ and $2(1-p)$, respectively. At the second stage (Figure 8c), I_0 is split into two equal-length subintervals I_{00} = [0, 1/4] and I_{01} = [1/4, 1/2] and the masses $m_{00} = p^2$ and $m_{01} = p(1-p)$ are spread uniformly between them; I_1 is split into two equal-length subintervals I_{10} = [1/2, 3/4] and I_{11} = [3/4, 1] and the masses $m_{10} = p(1-p)$ and $m_{11} = (1-p)^2$ are spread uniformly between them; The density on I_{00}, I_{01}, I_{10} and I_{11} is $4p^2$, $4p(1-p)$, $4p(1-p)$ and $(1-p)^2$, respectively. This construction continues recursively. Formally, at stage n, $n \in N$, each interval $I_{\varepsilon_1\varepsilon_2\cdots\varepsilon_{n-1}}$ in stage $n-1$ is split into two equal-length subintervals $I_{\varepsilon_1\varepsilon_2\cdots\varepsilon_{n-1}\varepsilon_n}$ with mass $m_{\varepsilon_1}m_{\varepsilon_2}\cdots m_{\varepsilon_{n-1}}m_{\varepsilon_n}$, $\varepsilon_i = 0, 1$. Therefore, $\mu(I_{\varepsilon_1\varepsilon_2\cdots\varepsilon_n}) = m_{\varepsilon_1}m_{\varepsilon_2}\cdots m_{\varepsilon_n}$. This defines a sequences of measures μ_n on the unit interval I, which converge weakly towards a probability measure μ, the binomial measure. From the procedure of construction, we can see that μ is strictly self-similar, as shown in Figure 9.

We can extend this construction to randomize the allocation of the mass in the recursive subdivisions. In this case, we may randomly choose the left multiplier as m_0 or m_1 (each with probability $= 0.5$), instead of always choosing m_0.

4.2 Property of Burstiness

Self-similar processes do not always generate bursty time sequences. Roughly speaking, the Hurst coefficient H describes global burstiness. However, local burstiness in disk I/O is more interesting in practice. Multifractals can represent local burstiness, as described by the local Hölder exponent and multifractal spectrum of binomial measures.

For any $x \in [0,1)$, there is a unique subinterval $I_{\varepsilon_1\varepsilon_2\cdots\varepsilon_n}$ containing it in stage n. Let us denote it as $I^{(n)}(x)$. For convenience, we let $m_0 > m_1$. For some x, the density on $I^{(n)}(x)$, $\mu(I^{(n)}(x))/|I^{(n)}(x)| = m_{\varepsilon_1} m_{\varepsilon_2} \cdots m_{\varepsilon_n}/2^{-n}$, tends to infinity when $n \to \infty$, as shown by the points in the leftmost subinterval in Figure 8 and Figure 9, where $\mu(I^{(n)}(x))$ is the mass on $I^{(n)}(x)$ and $|I^{(n)}(x)|$ is the length of $I^{(n)}(x)$. The coarse graining in this interval has the property of burstiness. We can use a singularity exponent, the Hölder exponent, $\alpha(x)$ as defined in the following equation to describe how fast the value approaches infinity:

$$\begin{aligned}
\alpha(x) &= \lim_{n\to\infty} \alpha^{(n)}(x) \\
&= \lim_{n\to\infty} \frac{\log_2 \mu(I^{(n)}(x))}{\log_2 |I^{(n)}(x)|} \\
&= \lim_{n\to\infty} \frac{\log_2 m_{\varepsilon_1} m_{\varepsilon_2} \cdots m_{\varepsilon_n}}{\log_2 2^{-n}} \\
&= -\lim_{n\to\infty} \frac{\log_2 \prod_{i=1}^{n} m_{\varepsilon_i}}{n}. \qquad (1)
\end{aligned}$$

The multifractal spectrum $f(\alpha)$ describes the global distribution of Hölder exponent $\alpha(x)$, which is defined in the following equation:

$$\begin{aligned}
f(\alpha) &= \lim_{n\to\infty} f^{(n)}(\alpha) \\
&= \lim_{n\to\infty} \frac{\log_2 N^{(n)}(\alpha)}{n}, \qquad (2)
\end{aligned}$$

with $N^{(n)}(\alpha)$ denoting the number of subintervals $I^{(n)}$ with Hölder exponent value of α.

At stage n, $\frac{n!}{i!(n-i)!} (= N^{(n)}(\alpha))$ subintervals have the same mass of $m_0^{n-i} m_1^i$. Therefore,

$$\begin{aligned}
\alpha^{(n)} &= -(\log_2 m_0^{n-i} m_1^i)/n \\
&= -(i/n)\log_2 m_1 - (1-i/n)\log_2 m_0 \qquad (3) \\
&= (i/n)\alpha_{max} + (1-i/n)\alpha_{min}, \qquad (4)
\end{aligned}$$

with $\alpha_{min} = -\log_2 m_0 = -\log_2 p$ and $\alpha_{max} = -\log_2 m_1 = -\log_2(1-p)$. According to Stirling's formula,

$$\frac{n!}{i!(n-i)!} \sim (2^{-n})^{-E(i/n)}, \text{ where } E(i/n) = -(i/n)\log_2(i/n) - (1-i/n)\log_2(1-i/n). \qquad (5)$$

Combining above equations, we can find

$$f(\alpha) = -\frac{\alpha_{max}-\alpha}{\alpha_{max}-\alpha_{min}} \log_2\left(\frac{\alpha_{max}-\alpha}{\alpha_{max}-\alpha_{min}}\right) - \frac{\alpha-\alpha_{min}}{\alpha_{max}-\alpha_{min}} \log_2\left(\frac{\alpha-\alpha_{min}}{\alpha_{max}-\alpha_{min}}\right), \alpha_{min} \le \alpha \le \alpha_{max}. \qquad (6)$$

This function has the same form as the entropy function, which provides us a way to estimate m_0 (bias p).

5 Multifractal I/O Request Synthesis

Multifractals represent locally bursty I/O behavior more accurately than other means of generating self-similar traffic. Here we introduce a method to use multifractals to model I/O request interarrival times at small timescales.

5.1 Estimation of Bias

The parameter bias p in binomial multifractals (or m_0 in binomial measures) describes the local burstiness behavior, which can be estimated from the real traces. There are several ways to estimate the bias p and we only introduce the two we used in our experiments.

The first way to estimate p is from the multifractal spectrum $f(\alpha)$ of binomial multifractals. We know that $f(\alpha)$ has the same shape as an entropy function. The bias p determines the location of the curve and how it is stretched. We can find the best fitting bias by visually judging how well the practical curve fits the theoretical ones.

The second way to estimate p is from the entropy value. Wang *et al* [17] proposed to use the entropy value of real traces to estimate the bias because of its reliability and efficiency.

Assume that S is an information source that emits independent symbols from alphabet $\{s_0, s_1, \cdots, s_{k-1}\}$ with probabilities $\{p_0, p_1, \cdots, p_{k-1}\}$, respectively ($\sum p_i = 1$). The average amount of information we obtain by observing the output of S is called *entropy* [15] and is defined as

$$E(p_0, \cdots, p_{k-1}) \quad = \quad -\sum_{i=0}^{k-1} p_i \log_2 p_i. \tag{7}$$

The disk traces can be viewed as a discrete time sequence Y_t, whose length can be normalized to be 1. For the purpose of model fitting, we can aggregate it at level n:

$$Y_t^{(n)}(k) \quad = \quad \int_{k2^{-n}}^{(k+1)2^{-n}} Y_t dt, \text{ where } k = 0, 1, \cdots, 2^n - 1. \tag{8}$$

At level n, the sequence $Y_t^{(n)}$ can be considered as a distribution of an information source with alphabet $\{s_0, s_1, \cdots, s_{2^n-1}\}$, whose entropy is given by

$$E_p^{(n)} \quad = \quad -\sum_{k=0}^{2^n-1} \frac{Y_t^{(n)}(k)}{\int_0^1 Y_t dt} \log_2 \frac{Y_t^{(n)}(k)}{\int_0^1 Y_t dt}. \tag{9}$$

If we plot the value $E_p^{(n)}$ against n, as proved in [17], the points should form a line with slope $E_p^{(1)}$ for a self-similar process like binomial multifractals. Thus, we can estimate $E_p^{(1)}$ from these points and the bias p using Equation 10:

$$E_p^{(1)} \quad = \quad -p \log_2 p - (1-p) \log_2 (1-p). \tag{10}$$

5.2 Verification of Estimation of Bias p

Not every I/O trace can be fit to a multifractal distribution. We empirically qualify the necessary characteristics of an I/O trace to accurately estimate bias p.

We estimate bias p by dividing the each interval into x bins, aggregating the requests within each bin, and using the entropy value and Equation 10. Choice of an appropriate bin size is crucial to the success of the method; if it is too large, bursty requests are aggregated, destroying the burstiness. If the bin size is too small, the fraction of empty bins is too large and there are not enough samples to estimate p. In intervals with little I/O activity, it may simply not be possible to estimate p.

We know from §3 that to keep the error under 5%, we should choose an interval between 3–10 seconds. We also need to choose a bin size that yields a reasonable percentage of non-empty bins without overaggregating. Figure 10 shows a typical cumulative distribution functions of interarrival times from cello and snake. We see that the percentage of requests with an interarrival time of < 10 ms is 15% for cello and 20%

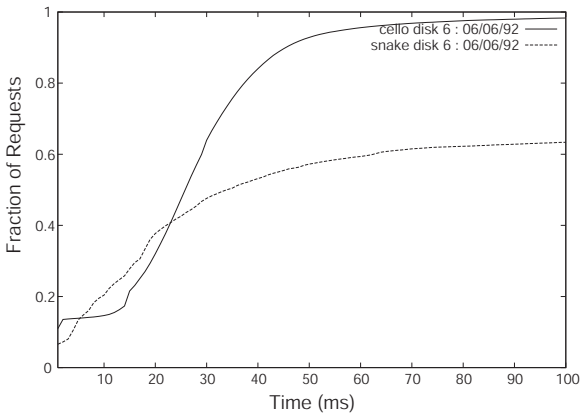


Figure 10: Cumulative distribution functions of interarrival times.

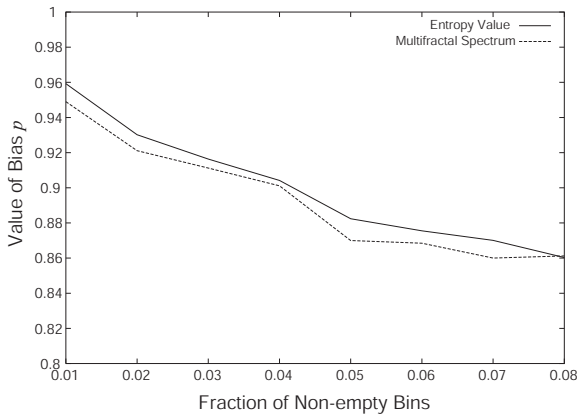


Figure 11: Estimation of p using entropy method and spectrum method.

for snake; these percentages are relatively small. Based on this observation, we select a bin size of 10 ms to avoid overaggregating requests. We choose an interval size of 5.12 seconds so that the number of bins within an interval is a power of 2.

To determine what fraction of non-empty bins is necessary to obtain a good estimate of p, we calculate p using both the entropy method and the spectrum method for selected data sets with certain percentages of non-empty bins, as shown in Figure 11. We could not exhaustively test all the data, because estimation of p using the multifractal spectrum is a visual test. Therefore, we selected a subset of datasets as follows. Because data sets with the same percentage of non-empty bins might have different aggregation ratios (number of requests in the interval / number of non-empty bins), we used the histogram of aggregation ratios to further round out our sample data sets. For example, if 20% of the intervals for a trace with 4% non-empty bins have an aggregation ratio of 1.1 (rounding to the nearest tenth), 20% of our samples have those characteristics.

We require that the fraction of non-empty bins be at least 3% for p to be meaningful. If the fraction of non-empty bins is smaller than 3% we can use any distribution, for example, a uniform distribution, to fit the data.

In summary, to accurately estimate bias p, we use a bin size of 10 ms, an interval size of 5.12 seconds, and consider p to be meaningful only when at least 3% of the bins are non-empty.

5.3 Multifractal Interarrival Synthesis Algorithm

We propose a new algorithm for synthesizing interarrival patterns based on a real trace. The approach is to fit intervals of a trace to a multifractal distribution, calculating p using Equation 10, as shown in Figure 12.

The key idea of synthesis is to use the request volume (mass) from the original trace and redistribute the mass in time according to the calculated bias. Figure 13 shows our algorithm for multifractal trace generation, based on [17]. This algorithm is improved as follows. Note that the original algorithm distributes mass at units as small as 1KB and creates many small requests, which can induce a synthetic error as high as 800%. To avoid this, we use the knowledge that the size of 70–80% of disk requests in the cello and snake traces, generated under HP-UX, are 8KB [13] and define the common request size r ($r = 8$KB) as an input to the algorithm.

Figure 14 shows how to use IMPROVED-BINOMIAL-MULTIFRACTAL-GENERATION to synthesize a trace. This algorithm takes as input a selected interval length s and a bin size b selected as described in §5.2, the common request size r as described above, and the original trace (from which we calculate p).

CALCULATE-P
INPUT: length l, trace interval w
OUTPUT: bias p
ALGORITHM:

for each i from 1 to $log_2 l$
    calculate the entropy value $E^{(i)}$ of w using Equation 9
    $array[i] \leftarrow E^{(i)}$
end for
estimate bias p from entropy values in *array* using linear regression
return p

Figure 12: Bias p estimation algorithm.

IMPROVED-BINOMIAL-MULTIFRACTAL-GENERATION
INPUT: bias p, length l, initial mass $sum(m)$, common request size r
OUTPUT: a binomial multifractal $((t_1, m_1), (t_2, m_2), \ldots, (t_n, m_n))$.
ALGORITHM:

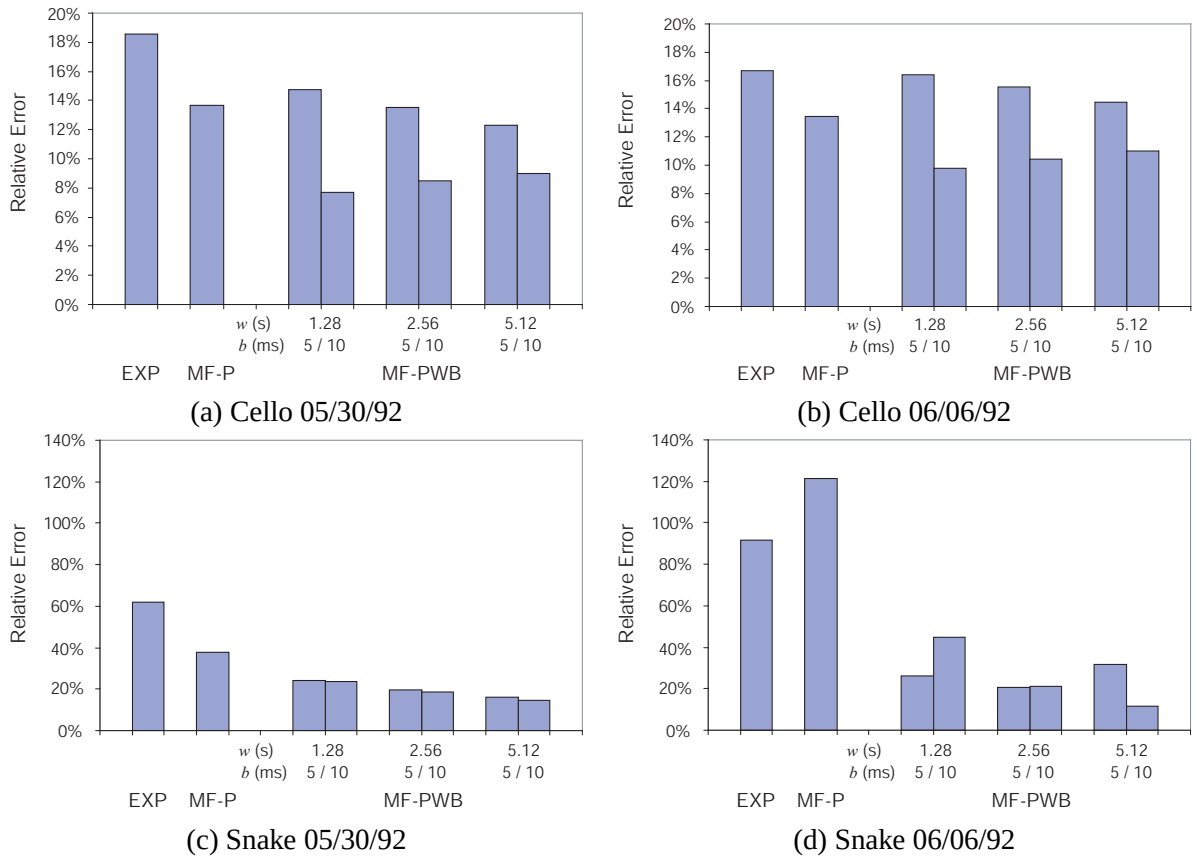
1. Initialize the stack and push pair (l, m) onto the stack.
2. If the stack is empty, return. Otherwise, go on to Step 3.
3. Pop a pair (l_i, m_i) from the stack. If $l_i = 1$, distribute the mass m_i in requests of size r and then go back to Step 2; if $0.5 * r < m_i < 1.5 * r$, try to combine the the top item in the stack to generate output and then go back to Step 2.
4. Flip a coin. If head, push pairs $(l_i/2, m_i * p)$ and $(l_i/2, m_i * (1-p))$ into the stack; if tail, push them in reverse order. Go back to Step 2.

Figure 13: Binomial multifractal I/O request generation.

SYNTHETIC-TRACE-GENERATION
INPUT: interval length s, bin size b, original trace file f, common request size r
OUTPUT: synthetic trace file
ALGORITHM:

for each non-empty interval w in f
    if fraction of non-empty bins $< 3\%$, $p = 0.5$
    else p = CALCULATE-P(s/b, w)
    resolution = 1 ms
    length = s/resolution
    mass = volume of requests in interval
    IMPROVED-BINOMIAL-MULTIFRACTAL-GENERATION $(p, length, mass, r)$
    map local timestamps to real timestamps
end for

Figure 14: Synthetic I/O trace generation algorithm.



(a) Cello 05/30/92

(b) Cello 06/06/92

(c) Snake 05/30/92

(d) Snake 06/06/92

Figure 15: Relative error of interarrival and request size synthesis methods. Note that EXP generates synthetic interarrivals only.

6 Simulation Results

In this section, we analyze the accuracy of our proposed I/O trace generation method. Our approach is to use the algorithm proposed in Figure 14 to generate a synthetic trace. Because we do not attempt to synthesize sector numbers or read/write operations, we retain the same sector identifiers and operations from the original trace. Therefore, our baseline for comparison is the original trace. We use Pantheon as in §3 to compare the CDF of disk response times for the original and synthetic traces.

To measure the improvement obtained by using syntheses based on fine-grained trace parameters, we compare our method, MF-PWB, to a simple exponential interarrival model, EXP, and a variant of the method proposed by Wang *et al* [17], MF-P. In MF-P, p is calculated over the entire trace ($w = 1$ day), but we use the algorithm of Figure 13, adjusting the trace for the most common request size. The key difference between MF-P and MF-PWB is that MF-PWB uses more parameters and does not model long-range dependence at timescales greater than w. Because we do not synthesize sector numbers or read/write operations, we retain the same sector identifiers and operations from the original traces. This preserves the spatial locality.

Figure 15a–15d show the relative error of the response time distribution for two different days (05/30/92 and 06/06/92) for EXP, MF-P and MF-PWB with different parameters. We selected these specific days because they have the maximum and minimum mean response time for the snake traces, which in general are more bursty, and harder to model, than cello. We can see that the relative error for MF-PWB method ranges from 7.7% to 44.6%, depending on the trace itself and the parameters w and b, the error for MF-P

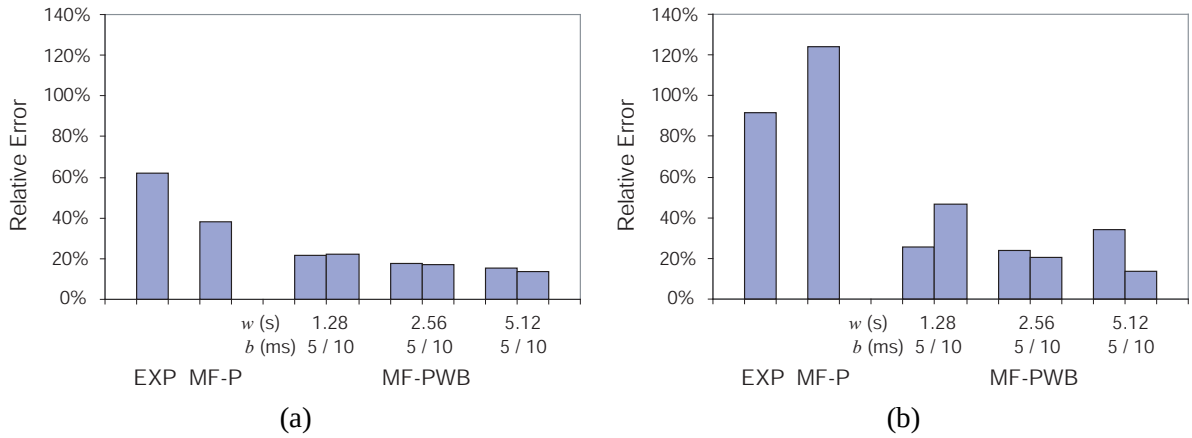


Figure 16: Relative error of interarrival synthesis for snake traces in (a) 05/30/92 and (b) 06/06/92.

ranges from 13.5% to 121.6%, and the error for EXP ranges from 16.7% to 91.6%, which is at least twice the error for MF-PWB.

In general, MF-PWB reproduces interarrival patterns more accurately than MF-P; computing p at smaller time intervals generally translates into more accurate synthesis. For snake traces, the improvement is significant. For cello traces, we can lose this effect by selecting poor values of b; however, the results are still comparable.

To better illustrate the quality of synthetic arrival times and request sizes, we isolate the effects of each. Figures 16a and 16b show the relative error of the response time distribution for snake traces (05/30/92 and 06/06/92) for EXP, MF-P and MF-PWB. Only the arrival times are synthetic; we obtain all other parameters from the original trace. The errors are almost the same as those from traces with synthetic arrival times and request sizes, as shown in Figures 15c and 15d.

Figures 17a and 17b show the relative error of the response time distribution for snake traces (05/30/92 and 06/06/92) for MF-P and MF-PWB. Here, only the request sizes are synthetic, and all other parameters are taken from the original traces. We do not compare EXP with the others because it does not generate synthetic request sizes. Request size synthesis accounts for less than 10% of the synthesis error, and MF-P is the most successful method for that component of synthesis.

Results for cello are similar [9]. The majority of synthetic error from MF-P and MF-PWB comes from synthetic arrival times.

7 Conclusions and Future Work

For purposes of performance evaluation, we need only consider I/O activity at timescales related to the system we are evaluating. For measuring disk response time and queuing behavior, we determined that a interval length of 5 seconds bounded the error to less than 5% for both the HP97560 and HP2204A disks under two different workloads.

However, accurately capturing burstiness is extremely important. We demonstrated a method of synthesizing interarrival times using binomial multifractals that exploits the fact that long-range dependence is unnecessary beyond certain timescales. Using this method, we synthesized traces with a relative error that ranged from approximately 8% to 12% on random and sequential workloads.

We are currently working on methods to automatically determine the appropriate interval length, and on combining this model for temporal locality with a similar one for spatial locality.

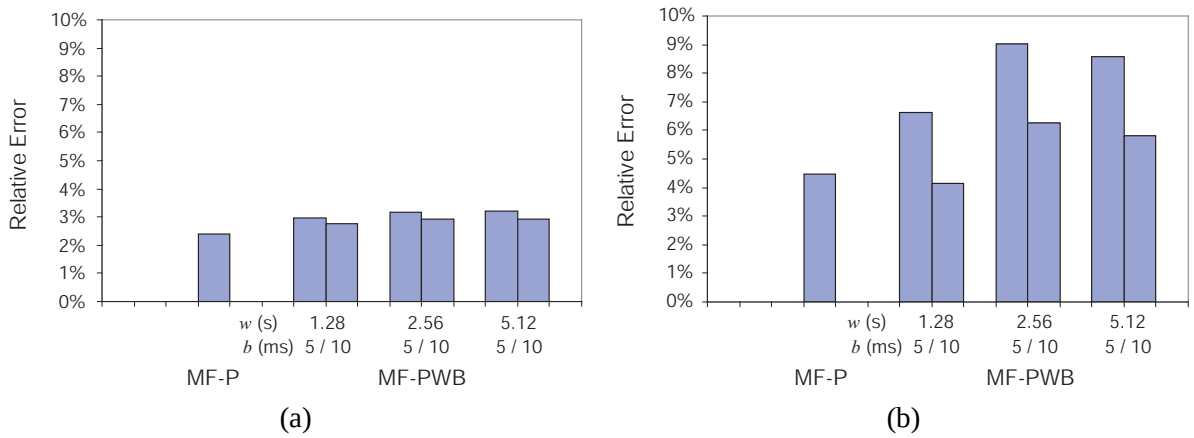


Figure 17: Relative error of request size synthesis for snake traces in (a) 05/30/92 and (b) 06/06/92.

Acknowledgements

We wish to thank Mengzhi Wang and Christos Faloutsos for their help with binomial multifractal synthesis. We are also grateful to John Wilkes and his group for his suggestions on this work and for providing us traces and the Pantheon software.

This work was supported in part by the National Science Foundation under grants NSF CCR-0093051 and NSF IDM-0083130.

References

[1] BERAN, J. *Statistics for Long-Memory Processes.* Chapman & Hall, New York, NY, 1994.

[2] CHEN, F.-M., MELLOR, J., AND MARS, P. On burstiness of self-similar traffic models. In *Proceedings of the European Conference on Networks and Optical Communications (NOC '96).*

[3] CHEN, H., CAI, H., AND LI, Y. The multifractal property of bursty traffic and its parameter estimation based on wavelets. In *Proceedings of IEEE TENCON '97. IEEE Region 10 Annual Conference* (Dec. 1997), vol. 2.

[4] CROVELLA, M., AND BESTAVROS, A. Self-similarity in world wide web traffic, evidence and possible causes. *Sigmetrics* (1996), 160–169.

[5] GANGER, G. R. Generating representative synthetic workloads: An unsolved problem. In *Proceedings of Computer Measurement Group* (1995), pp. 1263–1269.

[6] GÓMEZ, M. E., AND SANTONJA, V. Self-similarity in I/O workload: Analysis and modeling.

[7] GRIBBLE, S. D., MANKU, G. S., ROSELLI, D., BREWER, E. A., GIBSON, T. J., AND MILLER, E. L. Self-similarity in file systems. In *Proceedings of SIGMETRICS'98* (1998).

[8] GROSSGLAUSER, M., AND BOLOT, J. C. On the relevance of long-range dependence in network traffic. *IEEE/ACM Transactions on Networking 7,* 5 (1999), 629–40.

[9] HONG, B. Techniques for synthetic I/O workload generation. Tech. Rep. UCSC-CRL-02-16, University of California at Santa Cruz, 2002.

[10] LELAND, W. E., TAQQU, M. S., WILLINGER, W., AND WILSON, D. V. On the self-similar nature of Ethernet traffic (extended version). *IEEE Transactions on Networking* (1994), 1–15.

[11] NEIDHARDT, A. L., AND WANG, J. L. The concept of relevant time scales and its application to queuing analysis of self-similar traffic (or is Hurst naughty or nice?). In *Measurement and Modeling of Computer Systems* (1998), pp. 222–232.

[12] RIEDI, R. H. Introduction to multifractals. Tech. Rep. 99-06, Rice University, Sept. 1999.

[13] RUEMMLER, C., AND WILKES, J. Unix disk access patterns. In *Proc. of the Winter'93 USENIX Conference* (1993), pp. 405–420.

[14] RUEMMLER, C., AND WILKES, J. An introduction to disk drive modeling. *IEEE Computer 27*, 3 (Mar. 1994), 17–28.

[15] SHANNON, C. E., AND WEAVER, W. *Mathematical Theory of Communication*. University of Illinois Press, 1963.

[16] TAQQU, M. Statistical methods for long-range dependence. `http://math.bu.edu/people/murad/methods/index.html`. WWW.

[17] WANG, M., T.MADHYASTHA, CHAN, N., PAPADIMITRIOU, S., AND FALOUTSOS, C. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic,. In *Proceedings of 18th Internal Conference on Data Engineering* (2002).

[18] WILKES, J. The Pantheon storage-system simulator. Tech. Rep. HPL-SSP-95-14, Storage Systems Program, Computer Systems Laboratory, Hewlett-Packard Laboratories, Palo Alto, CA, May 1996.