Temporal Characterization of SPEC CPU2006 Workloads: Analysis and Synthesis

Qiang Zou
School of Computer Science
Southwest University
Chongqing 400715 China
qzou@swu.edu.cn

Jianhui Yue
Dept. of Elec. & Computer
University of Maine
Orono, ME 04469 USA
jyue@eece.maine.edu

Bruce Segee
Dept. of Elec. & Computer
University of Maine
Orono, ME 04469 USA
segee@eece.maine.edu

Yifeng Zhu
Dept. of Elec. & Computer
University of Maine
Orono, ME 04469 USA
zhu@eece.maine.edu

***Abstract*—SPEC CPU2006 benchmark suite has been extensively studied, with efforts focusing on the requirement understanding of memory workloads from the SPEC CPU2006 suite. However, characterizing SPEC CPU2006 workloads from a time dependence perspective has attracted little attention. This paper studies the auto-correlation functions of the arrival intervals of memory accesses in all SPEC CPU2006 traces, and concludes that correlations in memory inter-access times are inconsistent, either with evident correlations or with little and no correlation. Different with the studies focused on the prior suites, we present that self-similarity exists only in a small number of SPEC2006 workloads. In addition, we implement a memory access series generator in which the inputs are the measured properties of the available trace data. Experimental results show that this model can more accurately emulate the complex access arrival behaviors of real memory systems than the conventional self-similar and independent identically distributed methods, particularly the heavy-tail characteristics under both Gaussian and non-Gaussian workloads.**

I. Introduction

Accurately characterizing memory access behavior in computation-intensive workloads is essential to understanding the performance of the memory system. Researchers in both academia and industry have developed various benchmark suites, such as SPEC CPU benchmarks [1], [2], to test and evaluate the memory systems. However, on the one hand, for many benchmarks, it takes weeks or even months to complete a single run on cycle accurate execution-driven simulators such as M5 [3], and SimpleScalar [4]. On the other hand, a memory system has a large design space to be explored, such as transition control between different power states. Accordingly it becomes increasingly more challenging to run benchmarks multiple times in order to obtain comprehensive and fair evaluation. Synthetic benchmarks provide an improved methodology to speed up the evaluation process, and one of the most critical issues in designing a synthetic benchmark is to accurately characterize the memory access behavior.

In this paper, from a temporal dependence perspective, we analyze the correlations of inter-access times in twenty nine sets of memory traces collected in the SPEC CPU2006 benchmark suites, and show the necessity to further study the self-similarity in the minor workloads with the evident correlation of memory accesses. However, neither conventional self-similar nor independent identically distributed (IID) methods seem to be appropriate to characterize memory accesses in all SPEC CPU2006 workloads. Thus, based on the alpha-stable process, we propose and evaluate a statistical model to faithfully synthesize memory workloads. To the best of our knowledge, little research work conducted on this topic has been reported in the literature.

This paper makes the following three contributions:

- Our study shows that there are strong or evident correlations between inter-access times in a small number of SPEC2006 memory workloads. This suggests that further study of self-similarity is needed to deep understand the statistical phenomena of memory accesses, and rigorous statistical evidences are then presented to show that memory accesses exhibit the self-similar property.
- Correlation study shows that there is only slight and even no correlation between inter-access times in most SPEC2006 workloads. So, conventional self-similar models seem to be inappropriate to characterize memory accesses in these workloads.
- We propose a mathematical model based on the α-stable process to accurately synthesize the memory access series. Experimental results show that the synthetic traces generated by our model can more faithfully emulate the memory access behaviors than the compared conventional IID and self-similar methods.

The rest of this paper is organized as follows. Section II gives an overview of the SPEC2006 memory traces studied in this paper and summarizes related research works. Section III studies the correlation of inter-access times and discusses the necessity of studying self-similarity in some SPEC2006 workloads. Section IV presents the rigorous statistical evidence of self-similarity in SPEC2006 workloads. Section V proposes a statistical model to synthesize the memory access series and compares the workloads synthesized by the proposed model with real traces and other synthetic workloads. Section VI concludes this paper.

II. Background and Motivation

A. SPEC CPU2006 Benchmark Suites

SPEC CPU2006 are the standardized computation-intensive benchmark suites widely used in both academia and industry to

978-1-4673-4883-6/12/$31.00 ©2012 IEEE

TABLE I
PROCESSOR PARAMETERS

Parameter	Value
Frequency	2 GHz
Core	Alpha-like out-order
L1 I-cache	32 KB
L1 D-cache	32 KB
L2 Cache	2 MB
L2 Cache Line Size	64 Bytes

TABLE II
DRAM PARAMETERS

Parameter	Value
Frequency	667 MHz
tRP: Row Precharge time	12 ns
tRCD: Row active to row active delay	12 ns
tRAS : Row Activation time	27 ns
tCAS: Delay to access a certain column	12 ns
#Ranks per DIMM	2
Rank capacity	256 MB
#Banks per Rank	4
#Rows per Bank	16,384
#Columns per Row	1,024
Channel Width	8 Bytes
Row Buffer Management Policy	close page
Memory Scheduling Algorithm	FCFS

comprehensively and fairly evaluate the performance of CPUs, memory systems, and compiler techniques. These benchmarks are developed by using platform-neutral C/C++ or Fortran languages and thus they can run on a wide variety of computer architectures.

SPEC2006 benchmark suites include integer and floating-point benchmarks. Integer benchmark consists of twelve applications, i.e., *perlbench*, *bzip2*, *astar*, *mcf*, *gobmk*, *hmmer*, *sjeng*, *xalancbmk*, *h264ref*, *gcc*, *libquantum* and *omnetpp*. Floating-point benchmark includes seventeen applications, i.e., *cactusADM*, *gromacs*, *namd*, *povray*, *bwaves*, *calculix*, *gamess*, *GemsFDTD*, *lbm*, *leslie3d*, *milc*, *soplex*, *dealII*, *sphinx3*, *tonto*, *wrf* and *zeusmp*. The detailed description of each application is given in Ref. [5].

We have run all SPEC CPU2006 applications and collected the memory access trace of the SPEC2006 benchmark suites using an execution-driven processor simulator called M5 [3]. We have integrated a cycle-level DRAM simulator named DRAMsim [6] into M5 in order to accurately simulate the memory system. Table I and II show the parameters of the processor and Micron DDR2 memory [7] used in our simulation experiments, respectively.

In order to evaluate the performance of computer's memory system, some prior reaserch efforts have been focused on characterizing memory workloads [8], [9], including SPLASH-2 workloads on shared memory multiprocessors systems [10] and SPEC CPU benchmarks [11]. On the one hand, the presence of self-similarity in memory workloads had been presented, and a self-similar generator of memory references had been used to artificially generate request memory traces ten year ago in Ref. [10]. Li [11] had also studied the scaling properties of SPEC2000 integer benchmarks, and proposed an on-line program scaling estimator to capture the execution characteristics of large program in-flight. On the other hand, SPEC CPU2006 benchmark suites tend to be much more compute-intensive than SEPC CPU2000. This observation motivates us to revisit SPEC2006 workloads and consider the following key question: is it also self-similar for memory access behaviors in SPEC2006 workloads and appropriate to use a self-similar method to characterize memory accesses in memory-intensive SPEC2006 applications?

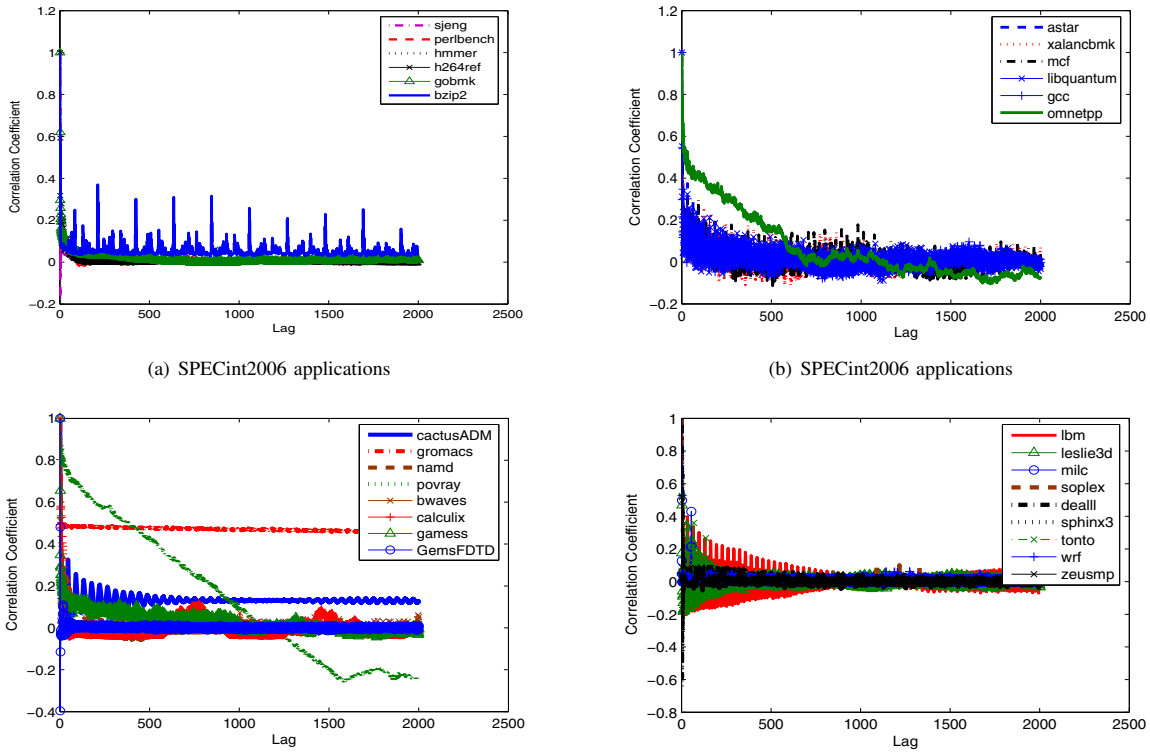
B. Related Work

Analysis of memory system access characteristics and patterns in various benchmarks such as commercial workloads [12], desktop applications [13], multimedia applications [14] and XQuery applications [15], has received considerable attention in the past few years. Several studies have investigated the basic characteristics of memory accesses, such as cache miss rates, memory intensity, and impacts of page size, in SPEC CPU benchmarks [16], [17], [18], [19]. Eeckhout *et al* [20] model the access sequence as a Statistical Flow Graph (SFG), in which basic blocks and their mutual transition probability are statistically identified. Joshi *et al* [21] and Bell *et al* [22] model memory accesses as a mixed sequence of constant and variable strides. Li [11] studies the scaling properties of SPEC2000 integer benchmarks and proposes an method to estimate the short-term and long-term execution characteristics of large programs. Sahuquillo *et al* [10] studies the self-similar properties of SPLASH-2 benchmarks, and constructs a self-similar memory reference generator which can flexibly makes a wide variety of workload traces.

The characteristics of self-similarity in data traffic was initially found in computer network traffic [23]. Since then extensively research work have been done to investigate this important nature in computer and network systems, such as the variable-bit-rate(VBR) video traffics [24], LAN [25], WAN [26] and web [27] traffics, file system [28] and disk-level [29], [30], [31], [32], [33] workloads.

Intensive research work has been done to study the characteristics of SPEC2006 workloads [34], [35], [19], [36], [17]. For example, based on microarchitecture-independent metrics such as the memory level parallelism (MLP), Ganesan *et al* [35] propose to extract the MLP from the real benchmark to estimate memory access burstiness, and build a model of the burstiness of memory accesses under the workloads of SPEC CPU2006 by considering the variations of the time intervals between consecutive burstiness of on-chip cache misses. However, no research has been done to study and examine the presence of self-similarity of memory accesses in SPEC CPU2006 workloads, to the best of our knowledge.

While self-similarity has been explored in the prior suite [11], in light of the more intensive burstiness in the SPEC CPU2006 workloads, this paper will examine the appropriateness of using the self-similarity to characterize memory access behaviors in the following section.



(a) SPECint2006 applications

(b) SPECint2006 applications

(c) SPECfp2006 applications

(d) SPECfp2006 applications

Fig. 1. Auto-correlation functions (ACFs) of memory accesses for the SPECint2006 and SPECfp2006 applications, respectively.

III. Correlation Study

In order to identify the statistical characteristics and gain a deep understanding of memory access behaviors, this section focuses on studying the correlation of inter-access times in memory access streams by using a widely used mathematical tool, called autocorrelation functions (ACF). The detailed introduction of this mathematical tool can be found in Ref. [37].

Given a set of observations $X = (X_t : t = 1, 2, \ldots, N)$, the correlation coefficient at $lag\ k$ is defined as

$$c_k = \frac{1}{N-k} \sum_{i=1}^{N-k} (X_i - \bar{X})(X_{i+k} - \bar{X}) \qquad (1)$$

where $\bar{X}$ is the expectation of the time series X. Then the auto-correlation function $ACF(k)$, with a lag of k, is defined as

$$ACF(k) = \frac{c_k}{c_0}. \qquad (2)$$

If the correlation coefficient of inter-access times quickly decreases to zero, it can be concluded that the memory access traffic is expected to be smooth instead of bursty and little or no correlations exist between the inter-access times. In this case it is reasonable to model the inter-access time as a sequence of random variables with independently and identically distribution (IID). On the contrary, if the correlation coefficient does not approach to zero quickly, then there exists some degree of correlations between inter-access times and such memory traffic is expected to be bursty instead of smooth. As a result, the inter-access time cannot be modeled as a simple IID random process and further study of auto-similarity is then necessary in order to correctly model the memory traffic.

In the following, we use auto-correlation functions (ACF) to study the characteristics in inter-access times for both the integer (SPECint2006) and floating-point (SPECfp2006) memory traces from a time dependence perspective. We present the analytical results of memory accesses in Figure 1.

Figure 1(a) and (b) respectively plot the auto-correlation coefficient of memory accesses for the studied integer benchmarks as the *lag* parameter increases gradually from 0 to 2000. As shown in Fig. 1(a)-(b), there are evident correlations for the memory inter-access times in *bzip2* and *omnetpp*. However, there is only a slight correlation for the memory inter-access times in *sjeng*, *perlbench*, *hmmer*, *astar*, *xalancbmk*, *mcf*, *gcc*, and *libquantum*, but only for $lag < 100$. And there is almost no correlation for the memory inter-access times in both *h264ref* and *gobmk*. The above observations indicate that it might be reasonable to further explore the existence of self-similarity in the *bzip2* and *omnetpp* workloads, but not for the *sjeng*, *perlbench*, *hmmer*, *astar*, *xalancbmk*, *mcf*, *gcc*, and *libquantum* workloads, especially for *h264ref* and *gobmk*.

In Fig. 1(c) and (d), we plot the auto-correlation coefficients

of memory inter-access times for the studied floating-point traces, as a function of *lag* from 0 to 2000, respectively. As shown in Fig. 1(c)-(d), there are strong correlations for both the auto-correlation functions of the memory inter-access times in both *gromacs* and *povray*. Especially, the correlations between inter-access times in *povray* include not only the positive correlation coefficients, but also the negative correlation coefficients. In addition, there are also evident correlations for the memory inter-access times in *cactusCDM*, *gamess* and *lbm*. However, as can be seen from Fig. 1(c)-(d), there is only a weak correlation for memory inter-access times in the remain floating-point memory traces, especially for *GemsFDTD*, and *sphinx3*, almost no correlation. Therefore, these observations suggest that the self-similar models seem to be a plausible option to model memory accesses in *gromacs*, *povray*, *cactusCDM*, *gamess* and *lbm*, but not for other memory workloads represented in Fig. 1(c)-(d).

We conclude that, in all SPEC2006 integer and floating-point memory traces studied, there is only slight and even no correlation between inter-access times in most SPEC2006 benchmarks. So, different with the studies focused on the prior suites in Ref. [11], [10], the independently and identically distributed (IID) but not self-similar property might be appropriate in characterizing memory access behaviors in these workloads.

However, there are strong or evident correlations in a small number of remaining traces, and it might still be appropriate to use the self-similarity to characterize memory access behaviors in these SPEC2006 workloads. This motivates us to examine the existence of self-similarity in SPEC2006 memory workloads with a rigorous statistical way in the following section.

IV. Self-similarity Study

In this section, we deploy Leland's theory and analysis techniques [23] to analyze and examine the existence of self-similarity in studied memory traces.

A. Theory of self-similarity

The theory behind self-similar processes is briefly summarized as follows. A more thorough description can be found in [23], [26], [25]. This section only outlines the basics that will be used in this paper. The description of self-similarity given below closely follows Beran *et al* [24].

Let $X = (X_t : t = 1, 2, \ldots)$ be a covariance stationary stochastic process with constant mean $\mu = E[X_t]$, and finite variance $\sigma^2 = E[(X_t - \mu)^2]$. For the process X_t, the autocorrelation function $ACF(k)$ depends only on k and is defined as follows.

$$ACF(k) = \frac{E[(X_t - \mu)(X_{t+k} - \mu)]}{E[(X_t - \mu)^2]},\ for\ k \geq 0. \quad (3)$$

The process X_t is said to exhibit self-similarity if

$$\lim_{k\to\infty} \frac{ACF(k)}{k^{-\beta}} = c < \infty,\ for\ 0 < \beta < 1. \quad (4)$$

Note that, in the equation above, $ACF(k)$ is non-sumable, i.e., $\sum_k ACF(k) = \infty$. We say that such an autocorrelation function decays hyperbolically and the corresponding process X_t is long-range dependent. In contrast, the autocorrelation function of a Poisson process decays exponentially and is sumable; that is $\sum_k ACF(k) = 0$. Such a process is said to be short-range dependent.

The process X_t is said to be exactly second-order self-similar with the *Hurst parameter* H $(0.5 < H < 1)$, if X_t has an autocorrelation function of the form

$$ACF(k) = \frac{1}{2}[(k+1)^{2-\beta} - 2k^{2-\beta} + (k-1)^{2-\beta}]. \quad (5)$$

For the process X_t, its m-order aggregated process $X^{(m)}$ is given as $X_k^{(m)} = \frac{1}{m}\sum_{j=0}^{m-1} X_{km-j}$, for $k \geq 1$. And

$$Var(X^{(m)}) = \sigma^2 m^{-\beta},\ for\ 0 < \beta < 1. \quad (6)$$

The process X_t is said to be asymptotically second-order self-similar with the Hurst parameter H $(0.5 < H < 1)$, if

$$\lim_{m\to\infty} ACF^{(m)}(k) = ACF(k) \quad (7)$$

The Hurst parameter noted H measures the self-similar degree of a time-series, and a value in the range $(0.5, 1)$ indicates self-similarity [38]. The larger the Hurst estimate is, the higher the degree of auto-similar property is. Two commonly used techniques are well-known graphical tools, namely *variance-time plots* [23], [28] and *R/S analysis* (Pox plot) [23], [28]. Both are widely used to judge whether self-similarity exists in a data traffic or not, and give the faithful examination results.

Variance-time plot. For an asymptotically second-order self-similar process X_t, the relation between the variance of the aggregated process $X^{(m)}$ and m is defined by Equation (6). Taking the logarithm of both sides of the equation results in the relation

$$log(Var(X^{(m)})) \approx a - \beta log(m), \quad (8)$$

where a is a constant, and $m \to \infty$ [28]. Thus, we can plot the curve of $log(Var(X^{(m)}))$ versus $log(m)$, for various values of m. The curve will be a linear series of points with slope $-\beta$, and using a linear regression method we can obtain an estimate of β. Slopes between -1 and 0 correspond to Hurst parameters H between 0.5 and 1. This plot is called a *variance-time plot*, and we can calculate the Hurst parameter H using the following equation

$$H = 1 - \frac{\beta}{2}. \quad (9)$$

R/S-Analysis. R/S (rescaled adjusted range) analysis, also called *Pox plot*. For a given set of observations $(X_t : t = 1, 2, \ldots, n)$ with a mean $\bar{X}(n)$, a variance $S^2(n)$, all observations are placed into K disjoint subsets, with each subset containing an average of n/K observations. Then the rescaled adjusted range statistic is given by [23]

$$\frac{R(n)}{S(n)} = \frac{1}{S(n)}[max(0, W_1, W_2, \ldots, W_n) - min(0, W_1, W_2, \ldots, W_n)]. \quad (10)$$

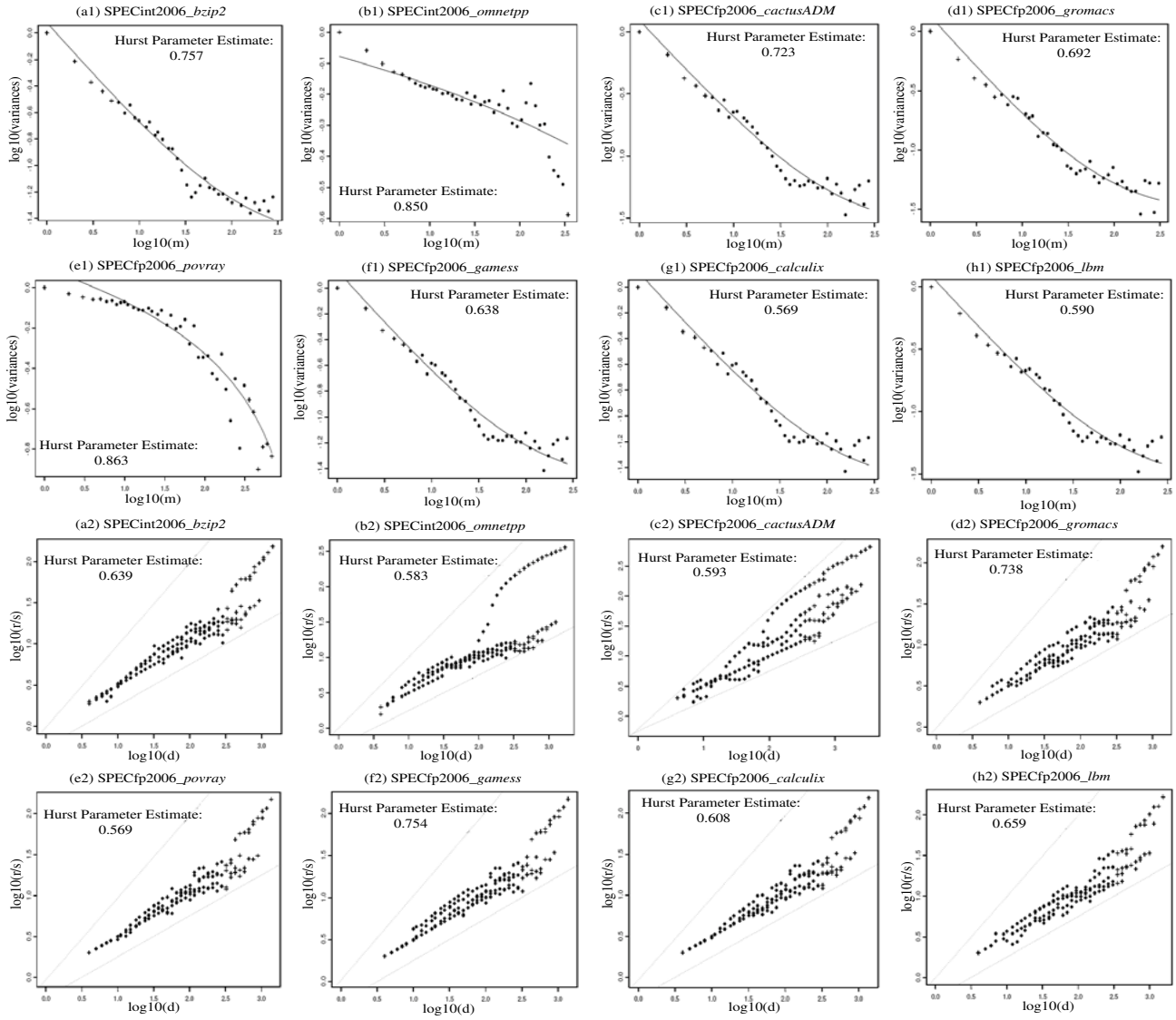


Fig. 2. Estimating the Hurst parameter: Plot (a1)-(h1) illustrate the variance-time plots, and Plot (a2)-(h2) figure the Pox plots for the integer and floating point benchmarks, respectively.

where $W_k = X_1 + X_2 + \ldots + X_n - k \cdot \bar{X}(n)$, for $k \geq 1$.

If X_t is self-similar or long-range dependent, then the following equation holds

$$E[\frac{R(n)}{S(n)}] \approx b \cdot n^H, \tag{11}$$

where $n \to \infty$, H is the Hurst parameter of X_t, and b is a constant. This empirical law is known as the *Hurst effect*.

Taking the logarithm of both sides of the equation results in the following relation

$$log(E[\frac{R(n)}{S(n)}]) \approx H \cdot log(n) + c, \tag{12}$$

where c is a constant, and $n \to \infty$. Thus we can plot $log(E[\frac{R(n)}{S(n)}])$ versus $log(n)$ for varying values of n, and obtain the estimate of the Hurst parameter H. This plot should result in a roughly linear graph with a slope equal to the Hurst parameter H. Such a plot is known as a *Pox plot*. So a least-squares linear fit can be used to estimate the Hurst parameter.

B. Measurement of Self-similarity

In this section, both the *variance-time plot* and *R/S-analysis* methods are used to estimate the Hurst parameter of memory access workloads, and mathematically demonstrate the presence of self-similar behavior in SPEC2006 workloads in which there are strong or evident correlations between inter-access times, i.e., *bzip2*, *omnetpp*, *cactusADM*, *gromacs*, *povray*, *gamess*, *calculix*, and *lbm*.

We generate the variance-time plots and Pox plots for all these traces above and then estimate their Hurst parameter

from the plots in Fig. 2. As shown in Fig. 2, Fig. 2(a1)-(h1) show the variance-time plots of the studied integer and floating-point SPEC2006 traces. All eight plots are linear and they have the Hurst parameter of 0.757, 0.850, 0.723, 0.692, 0.863, 0.638, 0.569 and 0.590, respectively. The results show that all Hurst parameters are significantly larger than 0.5. This verifies the presence of self-similarity in these memory access workloads.

Fig. 2(a2)-(h2) show the Pox plots of the same integer and floating-point benchmarks studied in Fig. 2(a1)-(h1). Following a least-square linear fit, the Hurst parameter is estimated as 0.639, 0.583, 0.593, 0.738, 0.569, 0.754, 0.608 and 0.659, respectively. All estimated Hurst parameters are also significantly larger than 0.5, indicating that the memory access behavior in these studied memory workloads are self-similar, which validates the results of Pox plot analysis and increases the confidence of the estimation accuracy.

The difference between the two measured *H* estimates for some integer memory traces (e.g., *omnetpp* and *povray*) is large, especially for *povray*. On one hand, this observation cannot be easily explained. Taking the R/S-Analysis estimate of *povray* as an example, the low value of the R/S-Analysis estimate (0.569) perhaps is a result of the existence of some regular memory accesses in *povray*, which causes the correlation coefficients of inter-access times fluctuate regularly in Figure 1(a). On the other hand, the difference verifies the common wisdom that there is no single estimator that can provide a definitive answer [39], although both the R/S-Analysis and variance-time plot methods can qualitatively demonstrate the existence of self-similarity.

In summary, both the R/S-Analysis and variance-time plot methods consistently confirm that the inter-access times of all SPEC2006 workloads studied in this section exhibit self-similarity. This indicates that the memory accesses in the SPEC2006 workloads in which there are strong or evident correlations between inter-access times tend to be very bursty, instead of smooth. If a model is required to characterize memory access arrivals, certainly a sequence of independently and identically distributed random processes is inappropriate.

However, the correlation studies in Section III show that a self-similar model might be inappropriate in characterizing memory access behaviors in most SPEC2006 workloads in which there is only slight and even no correlation between inter-access times. This motivates us to propose a more effective model to accurately characterize both the self-similar and IID SPEC2006 workloads in the next section.

V. Synthesizing Memory Workload Based on Alpha-stable Process

Previous sections have shown that memory access behaviors in SPEC2006 workloads exhibit independently and identically distributed or self-similar. In this section we presents a mathematical model that can be used to generate synthetically memory access workloads while preserving both the self-similar and IID properties.

A. Why use the alpha-stable?

Many techniques have been proposed to synthesize self-similar traffics [40], [41], [42], [28], [29], [38], [43], [44]. For example, two successful methods include Fractional Auto-Regressive Integrated Moving Average (FARIMA) and Fractional Brownian Motion (FBM). FARIMA [42] was first used to generate synthetic Variable Bit Rate (VBR) video traces. However, FARIMA is not intrinsically bursty. The FBM model used by several researchers [40], [41], [44] is easy to construct and can model the self-similarity under the Gaussian condition, but not the non-Gaussian condition. However, it is important to identify the Gaussian or non-Gaussian property for a given workload [45]. Otherwise, the real degree of access burstiness cannot be truthfully represented. In particular, when we synthesize the memory workloads, we also need to take the Gaussian property into considerations to avoid miss-presenting the burstiness.

For both integer and floating-point benchmarks, we use the normal quantile plots (QQ plots) to measure the Gaussian property [44]. The QQ plots of the *h264ref*, *xalancbmk*, *povray* and *cactusADM* traces, given in Figure 3, show that *xalancbmk* and *povray* are Gaussian, but others are non-Gaussian. In Figure 3(b) and (c), all of the scatter points corresponding to an access event given in the traces evidently follow a straight line, indicating that *xalancbmk* and *povray* are Gaussian. In Figure 3(a) and (d), all of the scatter points evidently don't fall into a straight line but an increasing curve, indicating that *h264ref* and *cactusADM* are non-Gaussian. The results above show interestingly that some memory workloads have the Gaussian property while other memory workloads do not. Therefore, the model used to capture the access burstiness in memory traces should be able to represent both the Gaussian and non-Gaussian properties. The α-stable process can meet this requirement well [45].

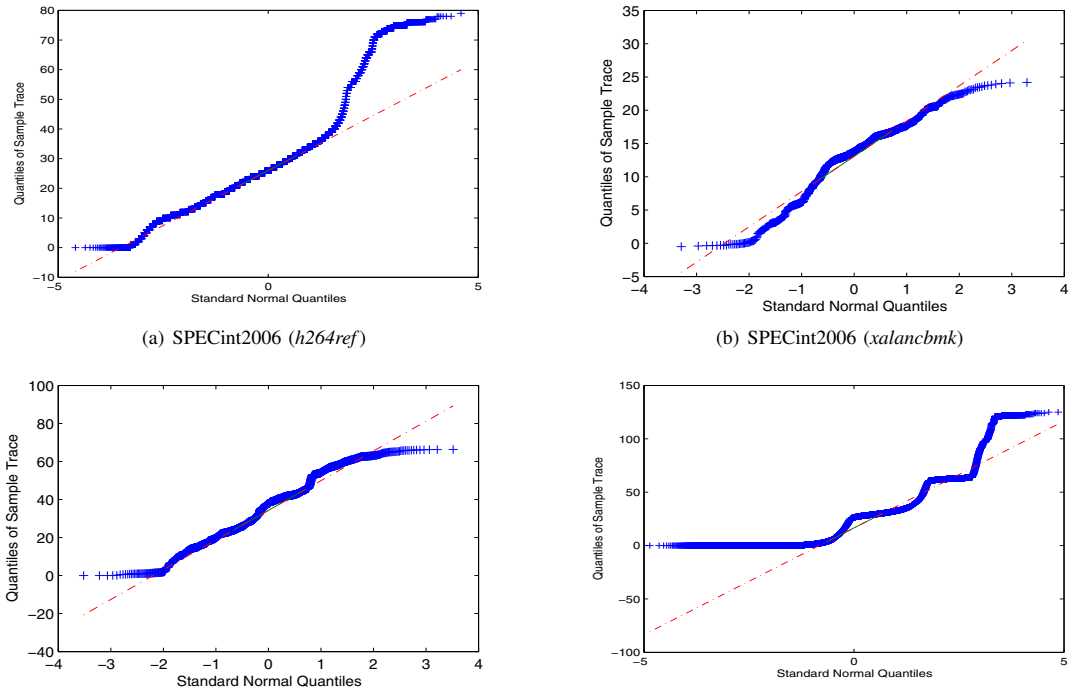
For a set of observations $X = (X_t : t = 1, 2, \ldots, n)$ with a mean μ, a variance $2\sigma^2$, the process X_t is said to be an alpha-stable process if its stable distribution is defined by its characteristic function [46]:

$$E[e^{i\theta X}] = \begin{cases} e^{-\sigma^\alpha |\theta|^\alpha (1 - i\beta sign\theta \tan\frac{\pi\alpha}{2}) + i\mu\theta}, & \alpha \neq 1 \\ e^{-\sigma|\theta|(1 + i\beta sign\theta \ln|\theta|) + i\mu\theta}, & \alpha = 1 \end{cases} \quad (13)$$

where $sign\theta$ is an indicative function, $0 < \alpha \leq 2$, $\sigma > 0$, $-1 \leq \beta \leq 1$, and $\mu \in R$. The characteristic exponent α measures the degree of burstiness in the memory workload, and β represents the degree of heavy tail in the memory workload.

If $\alpha = 2$, then $\beta \tan\pi = 0$ and β is then meaningless. In this case, it is the characteristic function of a Gaussian stochastic process, i.e., $E[e^{i\theta X}] = exp\{-\sigma^2\theta^2 + i\mu\theta\}$. Otherwise, it is one class of non-Gaussian functions. Therefore, as the value of parameter α changes, the α-stable process is able to flexibly represent a stochastic process under both the Gaussian and non-Gaussian conditions.

In this paper, we extend the α-stable model developed by Ref. [45] to synthesize memory access series. Specifically, the inputs in the α-stable model are the measured properties



(a) SPECint2006 (*h264ref*)

(b) SPECint2006 (*xalancbmk*)

(c) SPECfp2006 (*povray*)

(d) SPECfp2006 (*cactusADM*)

Fig. 3. Examine the Gaussian property of *SPECint2006* (e.g., *h264ref* and *xalancbmk*) and *SPECfp2006* (e.g., *povray* and *cactusADM*) workloads through QQ plots of sample data versus standard normal.

TABLE III
ESTIMATES OF THE PARAMETER OF α-STABLE DISTRIBUTION BASED ON MAXIMUM-LIKELIHOOD ESTIMATE.

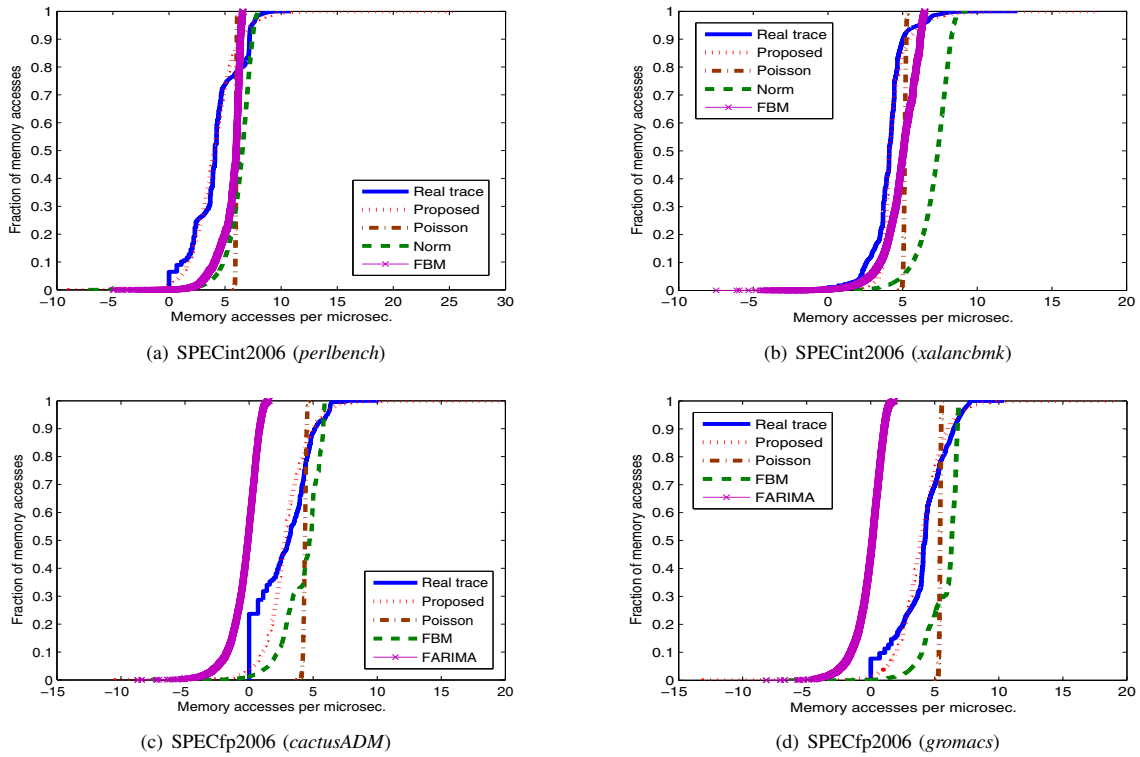
Trace type	Memory benchmarks	α-stable parameter			
		α	β	σ	μ
SPECint 2006	*perlbench*	0.725	1.00	33.3	28.33
	bzip2	1.195	1.00	26.41	36.70
	astar	1.402	0.395	50.27	19.8
	mcf	0.725	0.56	3.81	102.6
	gobmk	0.6756	0.6967	18.42	49.34
	hmmer	2.00	\	31.97	51
	sjeng	1.07	1.00	138.9	-110.6
	xalancbmk	2.00	\	10.58	73.2
	h264ref	0.7775	0.617	15.49	55.55
	omnetpp	0.62	0.49	10.84	20.17
	gcc	1.358	0.516	43.83	17.2
	libquantum	0.757	0.6492	26.27	70.1
SPECfp 2006	*cactusADM*	0.898	1.00	15.35	0.648
	gromacs	0.89	1.00	42.35	22.1
	namd	0.49	0.56	20.6	39.6
	povray	2.00	\	15804.6	9925
	bwaves	2.00	\	31.974	49
	calculix	0.6647	0.7582	24.0058	55.7115
	gamess	1.1879	1.00	1310.53	-397.19
	GemsFDTD	2.00	\	25.1599	15
	lbm	0.9025	1.00	4.0058	4.5759
	leslie3d	0.8906	1.00	3.5988	3.2536
	milc	2.00	\	30.4015	13
	soplex	0.9536	0.7578	11.6194	91.2489
	dealII	2.00	\	40.3606	69
	sphinx3	2.00	\	92.2529	84
	tonto	0.6972	0.6960	17.4512	44.9905
	wrf	0.6362	0.7121	17.1087	52.2860
	zeusmp	2.00	\	35.1190	54

of the available trace data, including the degree of self-similarity in the memory workload, the degree of memory access burstiness, the degree of heavy tail in the memory workload. This model allows us to conveniently turn the memory workload model for different environments.

B. Experiment results

For each benchmark, we compute the access arrival rate, i.e., the number of accesses per time unit. We place all access arrival rates into a group of stochastic numbers. The maximum-likelihood method is used to estimate the parameters of the α-stable process corresponding to the memory traces needed to be measured. Table III summarizes the estimates of α-stable parameters. In Table III, each row includes the memory trace name and the estimates of four α-stable parameters. Due to the fact that the α-stable distribution degenerates to a Gaussian stochastic process if $\alpha = 2$, with mean value μ and variance $2\sigma^2$, β will be meaningless. Accordingly in Table III, a slash will fill in the place of β if $\alpha = 2$. This measurement results prove the validity of Figure 3(b) and (c) again, i.e., the *xalancbmk* and *povray* workloads are Gaussian.

Our model can faithfully emulate the burstiness of memory access activities in all studied benchmarks. We used the alpha-stable, IID (Poisson and Normal) and self-similar FBM methods to synthesize the IID SPEC2006 workloads, and used the alpha-stable, self-similar (FBM and FARIMA), and IID Poisson methods to synthesize the self-similar SPEC2006



(a) SPECint2006 (*perlbench*)

(b) SPECint2006 (*xalancbmk*)

(c) SPECfp2006 (*cactusADM*)

(d) SPECfp2006 (*gromacs*)

Fig. 4. Comparison of CDFs between synthetic memory trace and real trace for SPECint2006 (e.g., the IID *perlbench* and *xalancbmk*) and SPECfp2006 (e.g., the self-similar *cactusADM* and *gromacs*).

workloads. The *cumulative distribution functions* (CDFs) are used to intuitively compare the synthetic workloads through both our proposed and other methods and trace results. The cumulative distribution functions (CDFs) of the synthetic and real traces for *perlbench*, *xalancbmk*, *cactusADM* and *gromacs*, are illustrated in Figure 4, the X-axis shows the logscale of memory access numbers per microsecond, and the Y-axis denotes the percentage of the arrival rates. A point $(x; y)$ in the cumulative distribution curve indicates that $y\%$ of access rates are less than or equal to an arrival rate of x.

The memory workload synthesized by the α-stable model very closely matches the real trace data, especially for IID *perlbench* and self-similar *gromacs*, as shown in Figure 4. It is evident that it is difficult for the conventional IID and self-similar methods to accurately capture the memory access burstiness which can be precisely characterized by the α-stable method.

A quantitative approach to evaluate the improvement is to analyze the error. A *trimmed mean* [47] is widely used to measure the central tendency and it is less sensitive to outliers that are far away from the mean. A trimmed mean is calculated by discarding a certain number of highest and lowest outliers and then computing the average of the remaining measurements. Since statistically a trimmed mean is usually more resilient and robust than a simple average mean, we use the trimmed mean to evaluate the matching degrees between each real workload and its corresponding synthetic workload. The trimmed means of errors and comparison results are summarized in Table IV.

As can be seen from Table IV, the IID SPEC2006 workloads are difficult to be faithfully characterized by self-similar method, and vice versa. For almost all of traces studied in this paper, the trimmed mean of error between the real workload and the α-stable synthetic workload is minimum, with the exception in which the trimmed mean of error between the *hmmer* trace and the α-stable synthetic workload is 41.9, with the increase of 15 percent of 36.4, the minimum error between the *hmmer* trace and the Poisson synthetic workload. Nevertheless, comparing with the matching degree of the Poisson synthetic workload, the matching degree of the α-stable synthetic workload for the *hmmer* workload is still reasonably good.

As shown in Table IV, for IID *xalancbmk* and *perlbench*, the trimmed means of errors between the real trace and the synthesized workload through the Poisson method are 61.2, and 365.3, respectively, the trimmed means of errors between the real trace and the synthesized workload through the Normal method are 45.5, and 271.3, respectively, and the trimmed means of errors between the real trace and the synthesized workload through the α-stable model with these parameter values in Table III are 29.5and 159, respectively. Accordingly, our proposed model can reduce the trimmed mean of error of the Poisson models by 52% and 56%, respectively, and reduce

TABLE IV
THE TRIMMED MEANS OF ERRORS FOR THE SPEC2006 BENCHMARKS.

SPECint2006	Poisson	Proposed	Norm	FBM	FARIMA
perlbench	365.28	158.99	271.32	384.5	\
bzip2	55.26	32.57	\	43.8	150.1
astar	30.2	19.7	36.05	42.9	\
mcf	87.13	72.59	80.94	104.26	\
gobmk	160.81	64.75	98.3	158.4	\
hmmer	36.4	41.9	52.7	84.1	\
sjeng	501.1	276.7	352.2	459.8	\
xalancbmk	61.23	29.52	45.5	78.3	\
h264ref	115.5	44.2	57.2	128.7	\
omnetpp	57.26	40.81	\	52.9	94.7
gcc	62.4	56.05	71.83	102.2	\
libquantum	20.7	14.4	23.8	57.1	\
SPECfp2006	Poisson	Proposed	Norm	FBM	FARIMA
cactusADM	64.11	35.72	\	47.5	85.3
gromacs	183.01	115.01	\	133.9	206.1
namd	392.42	158.02	162.3	478.8	\
povray	149.47	143.64	\	148.7	267.8
bwaves	63.5	54.1	58.6	97.3	\
calculix	262.4	243.6	302.5	403.7	\
gamess	97.3	52.7	\	71.5	185.2
GemsFDTD	72.9	62.8	68.1	114.4	\
lbm	80.5	51.6	\	61.8	129.5
leslie3d	82.4	75.1	78.6	128.4	\
milc	168.5	142.9	191.1	251.5	\
soplex	159.8	157.6	184.7	320.4	\
dealII	126.5	106.4	152.9	294.2	\
sphinx3	64.4	42.5	50.8	81.6	\
tonto	80.3	73.6	104.3	138.7	\
wrf	40.4	28.2	35.9	78.2	\
zeusmp	110.3	80.7	93.4	149.5	\

the trimmed mean of error of the Normal models by 35% and 41%, respectively. So, the synthetic IID workloads generated by the α-stable method are more accurate than the synthetic workloads synthesized by the IID methods. For *cactusADM* and *gromacs*, the trimmed means of errors between the real trace and the synthesized workload through the FBM method are 47.5 and 133.9, respectively, the trimmed means of errors between the real trace and the synthesized workload through the FARIMA method are 85.3 and 206.1, respectively, and the trimmed means of errors between the real trace and the synthesized workload through the α-stable model with these parameter values in Table III are 35.7 and 115, respectively. Accordingly, our proposed model can reduce the trimmed mean of error of the FBM models by 25% and 14%, respectively, and reduce the trimmed mean of error of the FARIMA models by 58% and 44%, respectively. So, the synthetic self-similar workloads generated by the α-stable method are more accurate than the synthetic workloads synthesized by the self-similar methods.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we studied the self-similarity phenomena of the memory access in the widely used SPEC 2006 benchmark suites. We examine the auto-correlation functions of inter-access times for all of memory access traces collected in SPEC2006. Results show that there are evident correlations between memory accesses in a small number of the integer and floating-point benchmarks. Therefore, a sequence of independent and identically distributed random variables is inappropriate to characterize and model memory accesses in the minor SPEC2006 workloads. This motivates us to further study the self-similarity in those SPEC2006 workloads. We have used rigorous statistical techniques, including variance-time plot and R/S analysis (Pox plot), to show the presence of self-similar property and estimate the Hurst parameter of memory access traces. In our experiments, all estimated Hurst parameters are significantly larger than 0.5.

However, correlation studies show that correlations in memory inter-access times are inconsistent. While with evident correlations between inter-access times in a small number of SPEC2006 benchmarks, there is only slight and even no correlation between inter-access times in most SPEC2006 workloads which cannot be accurately characterize by self-similar model. As a result, when characterizing the memory workloads or designing synthetic benchmark to evaluate a memory system, the characteristics in SPEC2006 memory accesses, should be taken into consideration to correctly preserve or emulate the access burstiness.

In addition, based on the α-stable process, we implement a memory access series generator in which the inputs are the measured properties of the available memory trace series. Experimental results show that this model can faithfully capture the complex access arrival characteristics of memory workloads, particularly the heavy-tail characteristics under both Gaussian and non-Gaussian workloads.

One limitation of this study is that all traces studied ignore the spatial information such as close/open bank model, address mapping schemes. Our immediate future work is to collect and study SPEC2006 memory traces in both the temporal and spatial locality.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their helpful comments in reviewing this paper. This project is supported by the US National Science Foundation (under Grant No. CNS-1117032, EAR-1027809, IIS-091663, CCF-0937988, CCF-0737583, CCF-0621493), the National Key Technology R&D Program under Grant No. 2012CB318208, the Fundamental Research Funds for the Central Universities under Grant No. XDJK2012A006, and the Ph.D. Foundation of Southwest University under Grants No. SWU111015.

REFERENCES

[1] "Spec cpu2006 published results," *http://www.spec.org/cpu2006/results*.
[2] "Spec cpu2000 published results," *http://www.spec.org/cpu2000/results*.
[3] N. L. Binkert, R. G. Dreslinski, and L. R. Hsu, "The m5 simulator: Modeling networked systems," *IEEE Micro*, vol. 26(4), pp. 52–60.
[4] D. C. Burger and T. M. Austin, *The simplescalar tool set, version 2.0*. University of Wisconsin, Madison: Technical Report CS-TR-97-1342, 1997.
[5] J. Henning, "Spec cpu2006 benchmark descriptions," *ACM SIGARCH Computer Architecture News*, vol. 34(4), pp. 1–17, 2006.
[6] D. Wang, B. Ganesh, and N. Tuaycharoen, "Dramsim: a memory system simulator," *SIGARCH Computer Architecture News*, vol. 33(4), pp. 100–107, 2005.

[7] M. Inc., "Micron 512mb: Ddr2 sdram data sheet," *http://www.micron.com*.

[8] Y. Kim, M. Papamichael, and O. Mutlu, "Thread cluster memory scheduling: Exploiting differences in memory access behavior," in *Proceedings of the MICRO-43*, Atlanta, Georgia, December 2010.

[9] Y. Kim, D. Han, and O. Mutlu, "Atlas: A scalable and high-performance scheduling algorithm for multiple memory controllers," in *Proceedings of the HPCA-16*, Bangalore, India, January 2010.

[10] J. Sahuquillo, T. Nachiondo, and J. Cano, "Self-similarity in splash-2 workloads on shared memory multiprocessors systems," in *Proceedings of the 26th EUROMICRO*, Maastricht, The Netherlands.

[11] T. Li, "Using a multiscale approach to characterize workload dynamics," in *Proceedings of the Workshop on Modeling, Benchmarking and Simulation (MoBS)*, Madison, Wisconsin, June 2005.

[12] L. A. Barroso, K. Gharachorloo, and E. Bugnion, "Memory system characterization of commercial workloads," in *Proceedings of the 25th International Symposium on Computer Architecture (ISCA)*, Barcelona, Spain, June 1998.

[13] D. Lee, P. Crowley, J. Baer, and T. Anderson, "Execution characteristics of desktop applications on windows nt," in *Proceedings of the 25th International Symposium on Computer Architecture (ISCA)*, Barcelona, Spain, June 1998.

[14] Z. Xu, S. Sohoni, R. Min, and Y. Hu, "An analysis of the cache performance of multimedia applications," *IEEE Transactions on Computers*, vol. 53(1), pp. 20–38, 2004.

[15] H. Liu, R. Li, and Q. Gao, "Characterizing memory behavior of xml data querying on cmp," in *Proceedings of the Workshop for Computer Architecture Evaluation of Commerical Workloads (CAECW'08), in conjunctions with the 14th International Symposium on High Performance Computer Architecture (HPCA-14)*, Salt Lake City, Utah, 2008.

[16] J. Henning, "Spec cpu2000: Measuring cpu performance in the new millennium," *IEEE Computer*, vol. 33(7), pp. 22–27, 2000.

[17] A. Jaleel, "Memory characterization of workloads using instrumentation-driven simulation–a pin-based memory characterization of the spec cpu2000 and spec cpu2006 benchmark suites," *VSSAD Technical Report*, 2007.

[18] S. Sair and M. Charney, "Memory behavior of the spec cpu2000 benchmark suite," *IBM Thomas J. Watson Research Center Technical Report RC-21852*, Oct. 2000.

[19] D. Ye, J. Ray, and D. Kaeli, "Characterization of file i/o activity for spec cpu2006," *ACM SIGARCH Computer Architecture News*, vol. 35(1), pp. 112–117, 2007.

[20] L. Eeckhout, R. H. B. Jr., and B. Stougie, "Control flow modeling in statistical simulation for accurate and efficient processor design studies," in *Proceedings of the 31st International Symposium on Computer Architecture (ISCA)*, Munchen, Germany, June 2004.

[21] A. Joshi, L. Eeckhout, R. H. B. Jr., and L. K. John, "Performance cloning: A technique for disseminating proprietary applications as benchmarks," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC'06)*, San Jose, California, 2006.

[22] R. H. B. Jr., R. R. Bhatia, and L. K. John, "Automatic testcase synthesis and performance model validation for high performance powerpc processors," in *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS'06)*, Austin, Texas, March 2006.

[23] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, vol. 2, pp. 1–15, Feb. 1994.

[24] J. Beran, R. Sherman, M. S. Taqqu, and W. Willinger, "Long-range dependence in variable-bit-rate video traffic," *IEEE Transactions on Communications*, vol. 43, pp. 1566–1579, Mar. 1995.

[25] 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 Transactions on Networking*, vol. 5(1), pp. 71–86, 1997.

[26] V. Paxson and S. Floyd, "Wide-area traffic: The failure of poisson modeling," *IEEE/ACM Transactions on Networking*, vol. 3(3), pp. 226–244, 1995.

[27] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM Transactions on Networking*, vol. 5, pp. 835–846, 1997.

[28] S. Gribble, G. Manku, and E. Brewer, "Self-similarity in high-level file systems: Measurement and applications," in *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS/Performance98)*, Madison, Wisconsin, June 1998.

[29] M. Gomez and V. Santonja, "Self-similarity in i/o workload: Analysis and modeling," in *Proceedings of the 1st IEEE International Workshop on Workload Characterization (WWC'98)*, Dallas, Texas, 1998.

[30] S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda, "Characterization of storage workload traces from production windows servers," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC'08)*, Seattle, WA, September 2008.

[31] A. Riska and E. Riedel, "Disk drive level workload characterization," in *Proceedings of the 2006 USENIX Annual Technical Conference*, Boston, MA, June 2006.

[32] B. Hong and T. Madhyastha, "The relevance of long-range dependence in disk traffic and implications for trace synthesis," in *Proceedings of the IEEE Conference on Mass Storage Systems and Technologies (MSST'05)*, Monterey, California, April 2005.

[33] A. Riska and E. Riedel, "Long-range dependence at the disk drive level," in *Proceedings of the Third International Conference on the Quantitative Evaluation of Systems (QEST)*, University of California, Riverside, CA, September 2006.

[34] J. Lin, Y. Chen, and W. Li, "Memory characterization of spec cpu2006 benchmark suite," in *Proceedings of the Workshop for Computer Architecture Evaluation of Commerical Workloads (CAECW'08), in conjunctions with the 14th International Symposium on High Performance Computer Architecture (HPCA-14)*, Salt Lake City, Utah, 2008.

[35] K. Ganesan, J. Jo, and L. K. John, "Synthesizing memory-level parallelism aware miniature clones for spec cpu2006 and implantbench workloads," in *Proceedings of the 2010 International Symposium on Performance Analysis of Systems and Software (ISPASS)*, White Plains, NY, March 2010.

[36] W. Korn and M. S. Chang, "Spec cpu2006 sensitivity to memory page sizes," *ACM SIGARCH newsletter, Computer Architecture News*, March 2007.

[37] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar, "Synthesizing representative i/o workloads for tpc-h," in *Proceedings of the Tenth International Symposium on High Performance Computer Architecture (HPCA-10)*, Madrid, Spain, February 2004.

[38] M. Gomez and V. Santonja, "Analysis of self-similarity in i/o workload using structural modeling," in *Proceedings of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, College Park, Maryland, October 1999.

[39] T. Karagiannis, M. Faloutsos, and R. Riedi, "Long-range dependence: Now you see it, now you don't!" in *Proceedings of the GLOBECOM*, Taipei, Taiwan.

[40] Norros, "On the use of fractional brownian motion in the theory of connectionless networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 15, pp. 200–208, 1997.

[41] Z. Kurmas, K. Keeton, and K. Mackenzie, "Synthesizing representative i/o workloads using iterative distillation," in *Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Orlando, Florida, October 2003.

[42] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar vbr video traffic," in *Proceedings of the ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, London, UK, September 1994.

[43] M. Wang, T. Madhyastha, and et al., "Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic," in *Proceedings of the 16th International Conference on Data Engineering (ICDE)*, San Jose, California, February 2002.

[44] C. Stathis and B. Maglaris, "Modelling the self-similar behaviour of network traffic," *Computer Networks*, vol. 34, pp. 37–47, 2000.

[45] Q. Zou, D. Feng, Y. Zhu, and H. Jiang, "A novel and generic model for synthesizing disk i/o traffic based on the alpha-stable process," in *Proceedings of the 16th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Baltimore, Maryland, September 2008.

[46] G. Samorodnitsky and M. Taqqu, *Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance*. New York: Chapman and Hall, 1994.

[47] Z. J. Liu and et al., *Computational Science Technique and Matlab*. Beijing, P. R. China: Science Press, 2001.