Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080441

Revisiting the Existence of Self-Similar Property in SPEC CPU Workloads

Qiang Zou[1,*], *Yong Li*[2], *Yujuan Tan*[3] *and Xuesong Chen*[4]

[1] School of Computer and Information Science, Southwest University, Chongqing 400715, China
[2] School of Mathematics, Chongqing Normal University, Chongqing 400047, China
[3] School of Computer, Chongqing University, Chongqing 400044, China
[4] School of Applied Mathematics, Guangdong University of Technology, Guangzhou 510006, China

Received: 10 Aug. 2013, Revised: 12 Nov. 2013, Accepted: 13 Nov. 2013
Published online: 1 Jul. 2014

Abstract: This paper studies self-similarity of memory accesses in high-performance computer systems. We analyze the auto-correlation functions of memory access intervals in SPEC CPU workloads with different time scales and present the statistical evidence that memory accesses have self-similar behavior. For memory traces studied in our experiments, all estimated Hurst parameters are larger than 0.5, which indicate that self-similarity seems to be a general property of memory access behaviors. In addition, a self-similar model is proposed to generate memory access series and experimental results show that this model can faithfully emulate the complex access behaviors of real memory systems.

Keywords: Memory access, self-similar, statistical evidence

1 Introduction

Due to the importance of accurately characterizing memory access behavior in computation-intensive workloads, analysis of memory system access characteristics and patterns has received considerable attention in the past few years. The SPEC CPU benchmarks are widely used for performance measurement across various computer memory subsystems in both industry and academia [8]. Several studies have studied the basic characteristics of memory accesses, such as cache miss rates, and impacts of page size, in SPEC CPU benchmark [4,5]. Eeckhout *et al* [19] 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* [20] and Bell *et al* [21] model memory accesses as a mixed sequence of constant and variable strides. Ganesan *et al* [18] propose extracting the memory level parallelism (MLP) from the real benchmark to estimate memory access burstiness and they model the burstiness of memory accesses by considering the variations of the time intervals between consecutive burstiness of on-chip cache misses. However, none of these studies statistically examine in detail the burstiness of memory access requests, particularly the widely popular phenomena of random fluctuations in request arrival rates at different time scales.

In this paper, we study 14 sets of memory traces collected in the SPEC CPU benchmarks. We demonstrate the existence of self-similar phenomenon in memory workloads over different time scales. We analyze the correlations of inter-access times and study the self-similarity in memory workloads. To the best of our knowledge, little research work on this topic has been reported in the literature.

This paper makes the following three contributions:

–Our study shows that there are evident correlations between inter-access time intervals in traces collected in both the integer and floating-point benchmarks of SPEC CPU, and exceptionally strong correlations in some benchmarks. This suggests that further study of the self-similarity is needed to understand the statistical phenomena of memory accesses.

–We present the mathematical evidence to show that memory accesses exhibit self-similar behavior over different time scales through detailed analysis of both integer and floating-point types of memory accesses.

[*] Corresponding author e-mail: qzoucs@gmail.com

– We propose a mathematical model to accurately synthesize the memory access series, particularly the heavy-tail characteristics.

The rest of this paper is organized as follows. Section 2 describes related works and discusses the necessity of studying self-similarity in memory workloads through studying the correlation of inter-access times of SPEC CPU benchmarks. Section 3 presents the statistical evidence of self-similarity in memory workload. Section 4 proposes a self-similar model to synthesize the memory access series and compares the workloads synthesized by the proposed model with real traces. Section 5 concludes this paper.

2 Background

Intensive research work has been done to study the characteristics of memory workloads [5,18,23]. For example, based on microarchitecture-independent metrics such as the memory level parallelism (MLP), Ganesan *et al* [18] build a model of the burstiness of memory accesses under the workloads of SPEC CPU and ImplantBench by considering the frequency of on-chip cache misses. However, no research has been done to study or confirm the self-similar nature of memory access workloads, to the best of our knowledge.

2.1 Memory access traces

In this paper, we choose SPEC CPU2000 as our target workloads. SPEC CPU2000 is a standardized computation-intensive benchmark suite widely used in both academia and industry to 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. SPEC CPU2000 includes 11 integer applications and 14 float-point applications, and the detailed description of each application is given in Ref. [5]. In both academia and industry, SPEC CPU2000 is still widely used.

We have collected the memory access trace of SPEC CPU benchmark suite using a execution-driven processor simulator called M5 [2]. We have integrated the cycle level DRAM simulator DRAMsim [3] in M5 to accurately simulate the memory system. Due to long simulation time and large file size of memory traces, we only collect the memory accesses made during a period of 250 million instructions. In the float point benchmarks, all simulations perform fast forwarding of the first 5000 million instructions in order to get representative workloads.

The traces collected in this work record the memory accesses of ten integer benchmarks, including *gcc*, *vpr*, *twolf*, *perlbmk*, *vortex*, *gzip*, *mcf*, *parser*, *gap* and *bzip*, and four floating-point benchmarks, including *ammp*, *applu*, *apsi* and *galgel*. All memory access timestamps were recorded in nanoseconds.

In order to identify the statistical characteristics and gain a deep understanding of memory access behaviors, in this paper, we first study the similarity of inter-access times in memory streams over different time spans by using autocorrelation functions (ACF). The detailed introduction of this mathematical tool can be found in Ref. [14,15].

2.2 Correlation study

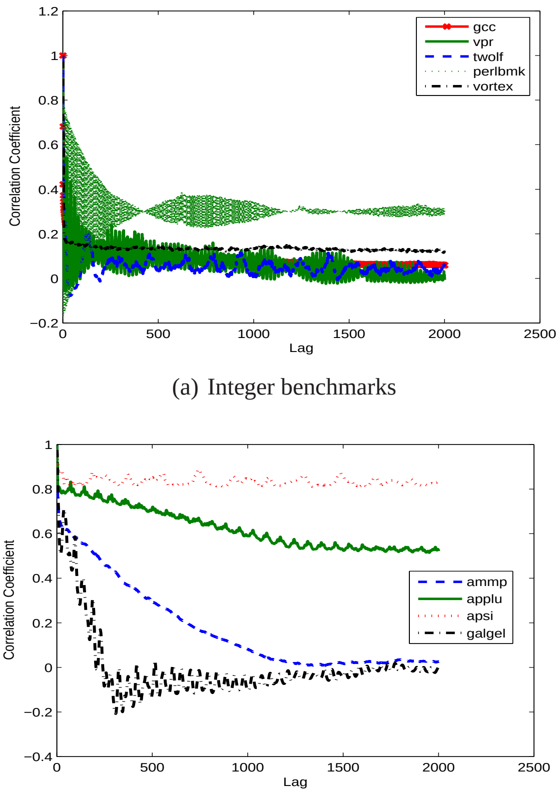
In the following, we use auto-correlation functions (ACF) to study the characteristics in inter-access times for both the integer and floating-point memory traces from a time dependence perspective. Due to space limitation, we only present the analytical results of *gcc*, *vpr*, *twolf*, *perlbmk*, *vortex*, *ammp*, *applu*, *apsi*, and *galgel*, as shown in Figure 1.

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.

Figure 1(a) and Figure 1(b) respectively plot the auto-correlation coefficient of memory accesses of studied integer benchmarks and floating-point benchmarks as the *lag* parameter increases gradually from 0 to 2000. The results show that there are evident correlations in all studied traces for all auto-correlation functions of the inter-access times, exceptionally strong correlations in some traces such as *applu* and *apsi*. This indicates that independently and identically distributed (IID) processes might not be appropriate in characterizing the memory accesses. Therefore, it is necessary to study and investigate the self-similarity of the memory traffic. This important observation motivates the research work of this paper.

3 Self-similarity in Memory Workloads

In this section, we present the statistical evidence to demonstrate the existence of self-similarity in studied memory traces.



(a) Integer benchmarks

(b) Floating point benchmarks

Fig. 1: Auto-correlation functions (ACFs) of memory accesses for the integer benchmarks (*gcc*, *vpr*, *twolf*, *perlbmk*, and *vortex*), and the floating point benchmarks (*ammp*, *applu*, *apsi*, and *galgel*).

The theory behind self-similar processes is briefly summarized as follows. A more thorough description can be found in [13]. This section only outlines the basics that will be used in Section V. The description of self-similarity given below closely follows Beran *et al* [12].

Let $X = (X_t : t = 1, 2, \dots)$ 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:

$$ACF(k) = \frac{E[(X_t - \mu)(X_{t+k} - \mu)]}{E[(X_t - \mu)^2]}, fork \geq 0. \tag{1}$$

The process X_t is said to exhibit self-similarity if

$$\lim_{k\to\infty} \frac{ACF(k)}{k^{-\beta}} = c < \infty, for 0 < \beta < 1. \tag{2}$$

Note that, in the equation, $ACF(k)$ is non-summable, i.e.,

$$\sum_k ACF(k) = \infty. \tag{3}$$

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 summable; that is $\sum_k ACF(k) = 0$. Such a process is said to be short-range dependent.

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 [13]. The larger the Hurst estimate is, the higher the degree of auto-similar property is. Two techniques that we employ are well-known graphical tools, namely *variance-time plots* (VTP) [12] and *R/S analysis* (Pox plot) [13], as discussed below.

Figure 3 illustrates the variance-time plots for the integer benchmarks (*gcc* and *vortex*), and the floating point benchmarks (*ammp* and *apsi*). The results show that all four plots are extremely linear and they have the Hurst parameter of 0.987, 0.995, 0.861 and 0.892, respectively. This verifies the self-similar nature of these memory access workloads. Especially, the curve in plot (a) and (b) are more linear than those in plot (c) and (d). This observation implies that there is a higher degree of self-similarity for memory traces *gcc* and *vortex*.

We generate the variance-time plots for all traces and then estimate their Hurst parameter from the plots, as shown in Table 1. The results show that all Hurst parameters are significantly larger than 0.5, indicating that all studied memory workloads exhibit self-similarity. It is surprising to find that the estimated Hurst parameters for all integer memory traces are approximately the same, such as 0.987 for *vortex*, 0.937 for *twolf*, 0.995 for *gcc* and 0.993 for *mcf*, and 0.996 for both *perlbmk* and *parser*. This indicates that these integer benchmarks have the similar level of self-similarity.

Table 1: Estimate of H using variance-time plot and Pox plot (R/S) for the integer and floating point benchmarks.

SPEC CPU	Benchmarks	Variance-time Plot	Pox Plot (R/S)
Integer	*gcc*	0.995	0.859
	vpr	0.872	0.699
	twolf	0.937	0.609
	perlbmk	0.996	0.507
	vortex	0.987	0.790
	gzip	0.906	0.795
	mcf	0.993	0.586
	parser	0.996	0.814
	gap	0.819	0.618
	bzip	0.994	0.786
Floating Point	*ammp*	0.861	0.653
	applu	0.629	0.696
	apsi	0.892	0.652
	galgel	0.934	0.716

Figure 4 shows the Pox plots of the same integer and floating-point benchmarks studied in Figure 3. Following a least-square linear fit, the Hurst parameter is estimated as 0.790, 0.859, 0.653 and 0.652 for *gcc*, *vortex*, *ammp*, and *apsi*, respectively. All estimated Hurst parameters are larger than 0.5, indicating that the inter-access time in

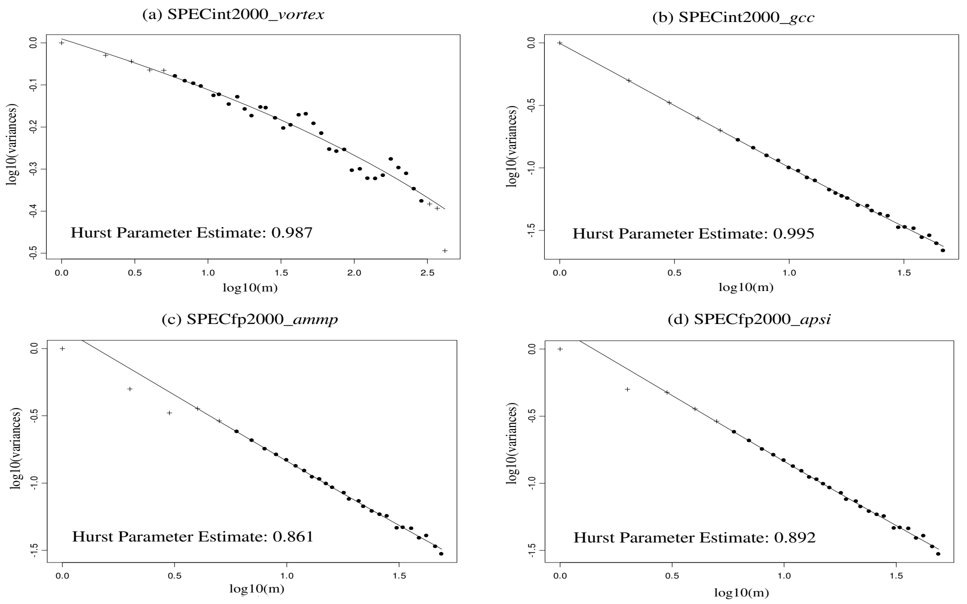


Fig. 2: Variance time plots for the integer benchmarks (*gcc*, *vortex*), and the floating point benchmarks (*ammp*, *apsi*). The variable *m* represents the aggregation level in microsecond.

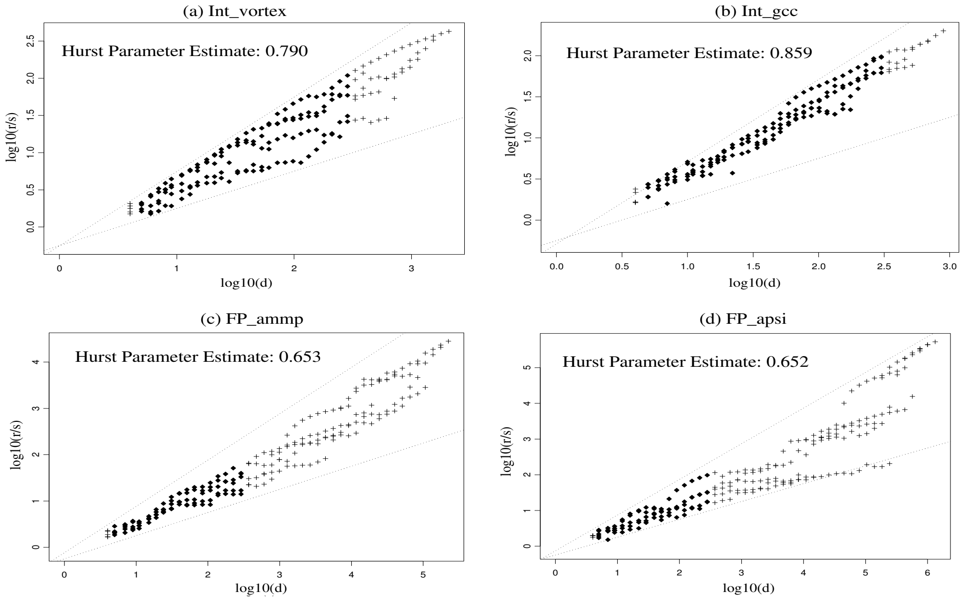


Fig. 3: Pox Plots for the integer benchmarks (*gcc*, *vortex*), and the floating point benchmarks (*ammp*, *apsi*). The variable *n* represents the size of non-overlapping block at which R/S statistic is computed.

these workloads are self-similar, which validates the results of Pox plot analysis and increases the confidence of the estimation accuracy.

We use the described R/S analysis technique to estimate the Hurst parameters of all memory traces that are collected in the SPEC CPU benchmark suite. The estimated Hurst parameters, listed in Table 1, are significantly larger than 0.5, confirming again the presence of self-similarity in the studied memory workloads.

The difference between the two measured *H* estimates for some integer memory traces (*twolf*, *mcf* and *perlbmk*) is large, especially for *perlbmk*. On one hand, this observation cannot be easily explained. Taking the

R/S-Analysis estimate of *perlbmk* as an example, the low value of the R/S-Analysis estimate (0.507) perhaps is a result of the existence of some regular memory accesses in *perlbmk*, 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 [17], although both R/S-Analysis and variance-time plots can qualitatively demonstrate the existence of self-similarity.

In summary, both the R/S-Analysis and variance-time plots consistently confirm that the inter-access times of all studied workloads exhibit self-similarity. This indicates that the memory accesses in the integer and floating-point benchmarks tend to be very bursty, instead of smooth. If a model is required to characterize memory accesses, certainly a sequence of independently and identically distributed random processes is inappropriate.

4 Synthesizing Memory Workloads

Previous sections have shown the statistical evidence to verify the existence of self-similar nature of memory accesses. In this section we presents a mathematical model that can be used to generate synthetically memory access workloads while preserving the self-similar property.

Many techniques have been proposed to synthesize self-similar traffics [13,12]. For example, the successful methods include Fractional Auto-Regressive Integrated Moving Average (FARIMA) and Fractional Brownian Motion (FBM). FARIMA [16] was first used to generate synthetic Variable Bit Rate (VBR) video traces. So, in this work, we used FARIMA model to synthesize SPEC CPU memory workloads, and compare the synthetic workloads through both our proposed and Poisson methods and trace 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.

Our model can faithfully emulate the burstiness of memory activities in all studied benchmarks. A quantitative approach to evaluate the improvement is to analyze the error. A *trimmed mean* [7] 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 2.

Table 2: The trimmed means of errors for the integer and floating point benchmarks.

SPEC CPU	Benchmarks	Poisson	Proposed	Improvement
Integer	*gcc*	61.22	57.28	6%
	vpr	6.79	6.43	5%
	twolf	3.04	0.65	79%
	perlbmk	8.20	5.04	38%
	vortex	98.51	95.86	3%
	gzip	4.81	2.18	54%
	mcf	4.74	4.63	3%
	parser	21.67	17.26	20%
	gap	51.84	46.94	9%
	bzip	16.35	14.10	14%
Floating Point	*ammp*	14.64	7.67	48%
	applu	19.38	12.75	34%
	apsi	18.06	15.32	15%
	galgel	17.65	11.93	32%

As shown in Table 2, for *bzip*, *parser*, *applu* and *apsi*, the trimmed means of errors between the real trace and the synthesized workload through the Poisson method are 16.35, 21.67, 19.38, and 18.06, respectively, and the trimmed means of errors between the real trace and the synthesized workload through the proposed model are 14.10, 17.26, 12.75, and 15.32, respectively. Accordingly, our proposed model can reduce the trimmed mean of error of the Poisson models by 14%, 20%, 34% and 15%, respectively.

The observations show that the memory workload synthesized by the proposed model very closely matches the real trace data, especially for *applu*. It is evident that it is difficult for the Poisson method to accurately capture the memory access burstiness which can be precisely characterized by the proposed method.. So, the synthetic workloads generated by the proposed method are more accurate than the synthetic workloads synthesized by the Poisson method.

5 Conclusions and Future Work

In this work, we studied the self-similar phenomena of the memory access in the widely used SPEC CPU benchmark suite. We examine the auto-correlation functions of inter-access times for all of memory access traces collected in SPEC CPU benchmarks. Results show that there are evident correlations between memory accesses in both the integer and floating-point benchmarks. Therefore, a sequence of independent and identically distributed random variables is inappropriate to characterize and model memory accesses. This motivates us to further study the self-similarity in memory workloads, which can provide useful insight into analysis of memory workloads, and design of memory benchmarks and synthetic workloads.

We have shown statistical evidence that the memory accesses are consistent with self-similar behavior in different time scales. We have used rigorous statistical techniques, including variance-time plot and R/S analysis (Pox plot), to estimate the Hurst parameter of memory access traces. In our experiments, all estimated Hurst parameters are significantly larger than 0.5, indicating that self-similarity seems to be a general property of memory access behaviors. As a result, when characterizing the memory workloads or designing synthetic benchmark to evaluate a memory system, the self-similarity, an intrinsic nature in memory accesses, should be taken into consideration to correctly preserve or emulate the burstiness.

In addition, we implement a self-similar model to synthesize memory access series and experimental results show that this model can faithfully capture the complex access characteristics of memory workloads, particularly the heavy-tail characteristics.

Acknowledgement

We would like to thank the anonymous reviewers for their helpful comments in reviewing this paper. This work is partially supported by the Ph.D. Foundation of Southwest University under Grants No. SWU111015, the Fundamental Research Funds for the Central Universities under Grant No. XDJK2012A006, the National Key Technology R&D Program under Grant No. 2012BAD35B08, the National Natural Science Foundation of China under Grant No. 61303038, and the Chongqing Natural Science Foundation under Grant No. CSTC2013jcyjA40050.

References

[1] M. Inc. Micron 512mb: Ddr2 sdram data sheet. http://www.micron.com.

[2] N. L. Binkert, R. G. Dreslinski, L. R. Hsu and et al. The m5 simulator: Modeling networked systems. IEEE Micro, **26**, 52-60 (2006).

[3] D. Wang, B. Ganesh, N. Tuaycharoen and et al. Dramsim: a memory system simulator. SIGARCH Computer Architecture News, **33**, 100-107 (2005).

[4] J. Henning. SPEC CPU2000: Measuring CPU Performance in the New Millennium. IEEE Computer, **33**, (2000).

[5] S. Sair and M. Charney. Memory Behavior of the SPEC CPU2000 Benchmark Suite. IBM Thomas J. Watson Research Center Technical Report RC-21852, (2000).

[6] Z. Xu, S. Sohoni, R. Min and Y. Hu. An Analysis of the Cache Performance of Multimedia Applications. IEEE Transactions on Computers, **53**, 20-38 (2004).

[7] Z. J. Liu. Computational Science Technique and Matlab. Science Press. Beijing, China, (2001).

[8] SPEC CPU2000 published results. http://www.spec.org/cpu2000/results.

[9] 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), (1998).

[10] D. Lee, P. Crowley, J. Baer and et al. Execution Characteristics of Desktop Applications on Windows NT. In Proceedings of the 25th International Symposium Computer Architecture (ISCA), (1998).

[11] H. Liu, R. Li, Q. Gao and et al. Characterizing Memory Behavior of XML Data Querying on CMP. In Proceedings of the Workshop for Computer Architecture Evaluation of Commerical Workloads (CAECW'08).

[12] J. Beran, R. Sherman, M. S. Taqqu, and W. Willinger. Long-range dependence in variable-bit-rate video traffic. IEEE Transactions on Communications, **43**, 1566-1579 (1995).

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

[14] J. Zhang, A. Sivasubramaniam, H. Franke and et al. Synthesizing Representative I/O Workloads for TPC-H. In Proceedings of the Tenth International Symposium on High Performance Computer Architecture (HPCA-10). Madrid, Spain, (2004).

[15] Q. Zou, and Y. Li. Transition Probability: A Novel Modeling Approach of Energy Consumption for Storage Subsystem. Applied Mathematics & Information Sciences, **7**, 371-377 (2013).

[16] 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, (1994).

[17] 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, (2002).

[18] 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, (2010).

[19] L. Eeckhout, R. H. Bell Jr., B. Stougie and et al. Control Flow Modeling in Statistical Simulation for Accurate and Efficient Processor Design Studies. In Proceedings of the 31st International Symposium on Computer Architecture (ISCA), (2004).

[20] A. Joshi, L. Eeckhout, R. H. Bell Jr., and Lizy K. John. Performance Cloning: A Technique for Disseminating Proprietary Applications as Benchmarks. In Proceedings of the IEEE International Symposium on Workload Characterization (IISWC), (2006).

[21] R. H. Bell Jr., R. R. Bhatia, L. K. John and et al. 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), (2006).

[22] D. C. Burger and T. M. Austin. The simplescalar tool set, version 2.0. Technical Report CS-TR-97-1342. University of Wisconsin, Madison, (1997).

[23] Y. Kim, M. Papamichael, O. Mutlu and et al. Thread Cluster Memory Scheduling: Exploiting Differences in Memory Access Behavior. In Proceedings of the MICRO-43, Atlanta, GA, (2010).

Qiang Zou, received his MS degree in applied mathematics and PhD degree in computer architecture, from Huazhong University of Science and Technology (HUST), Wuhan, China in 2005 and in 2009, respectively. He then worked as an Associate Professor in School of Computer and Information Science, Southwest University (SWU), China. His main research interests focus on workload characterization, Markov chain, storage system and performance evaluation. He has published several papers in journals and international conferences.

Yong Li, received his MS degree in applied mathematics and PhD degree in probability and statistics, from Huazhong University of Science and Technology (HUST), Wuhan, China in 2005 and in 2008, respectively. He then worked as an Assistant Professor in School of Mathematics Science, Chongqing Normal University, China. His current research interests include stochastic analysis and application, and stability theory.

Yujuan Tan, received her PhD degree in computer architecture, from Huazhong University of Science and Technology (HUST), Wuhan, China in 2012. She then worked as an Assistant Professor in College of Computer Science, Chongqing University, China. Her main research interests focus on data deduplication, data backup and recovery, storage system and parallel computing. She has published several papers in international conferences and journals.

Xuesong Chen, received his MS degree in computation mathematics from Huazhong University of Science and Technology, Wuhan, China, and PhD degree in control theory and control engineering from Guangdong University of Technology, Guangzhou, China, in 2004 and 2011, respectively. Now he works as an Associate Professor in school of applied mathematics, Guangdong University of Technology (GDUT), China. His current research interests include reinforcement learning, Markov chains, nonlinear control, intelligent control and their applications. He has published several papers in journals and international conferences.