Characterization of I/O Behaviors in Cloud Storage Workloads

Qiang Zou, Yifeng Zhu, Jianxi Chen, *Member, IEEE*, Yuhui Deng, and Xiao Qin

***Abstract*—As cloud platforms become increasingly popular, accurately understanding I/O behaviors in modern cloud storage is of paramount importance for system design and optimization. This paper sheds new light on the correlation of inter-arrival times of both read and write requests at the block level in four representative cloud storage workloads – AliCloud, Systor'17, MSRC and FIU. Our study reveals that I/O arrivals at the block level are very complex in modern cloud storage. There is a certain degree of correlation in the long-term timescale for request arrival intervals in AliCloud and Systor'17_read. Request arrival intervals in MSRC, FIU and Systor'17_write, however, are almost uncorrelated. The Gaussianity test confirms that I/O burstiness appears to be Gaussian in AliCloud_write and Systor'17_read, but the burstiness is non-Gaussian in other workloads. Importantly, we unfold the existence of self-similarity in cloud storage workloads with a certain degree of correlations, via visual evidence, the autocorrelation structure of the aggregated process of I/O request sequences, and Hurst parameter estimates. We further design an alpha-stable workload model for synthetic I/O generation, and the experimental results demonstrate that our model has an edge over conventional models in terms of accurately emulating I/O burstiness.**

***Index Terms*—Aggregated process, cloud block storage, inter-arrival time, self-similar, synthesis, workload characterization.**

I. Introduction

CLOUD block storage systems are critical infrastructure to render leading-edge cloud services, such as virtual desktops, database, web services, and key-value store [1], [2]. The design and optimization of cloud block storage systems are anchored on insights gained from block-level I/O workload studies. A viable first step in understanding I/O characteristics is to analyze inter-arrival time [1], [3], [4] followed by adopting a corresponding stochastic model to effectively characterize storage workloads [3].

On one hand, recent studies suggest that I/O burstiness widely exists in block-level workloads [5], [6], [7], [8], [9]. With increasing load intensity in cloud block storage systems [10], I/O activities in block workloads become more bursty [1] such that long-tailed properties are often observed [5], [11], [12]. On the other hand, previous studies often employed the traditional distributions including normal [11], [13], [14], [15], exponential [16], and Poisson [12], [17] to model I/O characteristics in storage workloads. Some studies even simply assume that I/O activities follow exponential distribution [16] or Poisson distribution [12], in favor of their attractive theoretical properties and modeling theories. In fact, it remains unexamined whether these traditional distributions can faithfully characterize the burstiness in modern cloud workloads. This fundamental and open question motivates us to revisit and delve in the understanding of block workloads in a modern production cloud environment.

We analyze throughout this paper four sets of block traces, which are collected in typical application environments, including *AliCloud* [1] gleaned at Alibaba, *MSRC* [18] offered by Microsoft Research at Cambridge, *Systor'17* [19] harvested at Fujitsu Laboratories Limited in Kawasaki, and *FIU* [20] from Florida International University. For these representative cloud block workloads, this study thoroughly examines the correlation of inter-arrival times of I/O requests and diagnoses the Gaussianity, diurnal rhythm, burstiness, and self-similarity in I/O patterns. In addition, we develop an I/O sequence generator based on a versatile model to synthesize workloads matching the temporal and spatial I/O behaviors. With good robustness, this novel generator is primed to be used to synthetically generate block I/O workloads for a diversity of application environments. To the best of our knowledge, little research work on production cloud block workload has been reported in the literature.

In summary, this study makes the following four contributions:

- We investigate the correlation of inter-arrival times of read requests and write requests in these four sets of representative block I/O workloads in modern cloud computing environments. We observe that there is a certain degree of correlation in the long-term timescale for request arrival intervals in AliCloud and Systor'17_read. In contrast,

Manuscript received 5 July 2022; revised 25 February 2023; accepted 26 March 2023. Date of publication 31 March 2023; date of current version 6 September 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62072214, in part by the Open Project of Key Laboratory of Ministry of Industry and Information Technology for Basic Software and Hardware Performance and Reliability Measurement under Grant HK202201317, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2021B1515120048. The work of Xiao Qin's was supported in part by the U.S. National Science Foundation under Grants IIS-1618669 and OAC-1642133. Recommended for acceptance by D.B. Whalley. *(Corresponding author: Yuhui Deng.)*

Qiang Zou is with the Department of Computer Science, Southwest University, Chongqing 400715, China (e-mail: qiangzou@hotmail.com).

Yifeng Zhu is with the Department of Electrical and Computer Engineering, University of Maine, Orono, ME 04469 USA (e-mail: yifeng.zhu@maine.edu).

Jianxi Chen is with the Wuhan National Laboratory for Optoelectronics, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: chenjx@hust.edu.cn).

Yuhui Deng is with the Department of Computer Science, Jinan University, Guangzhou 510632, China (e-mail: tyhdeng@jnu.edu.cn).

Xiao Qin is with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849-5347 USA (e-mail: xqin@auburn.edu).

Digital Object Identifier 10.1109/TC.2023.3263726

request arrival intervals are almost uncorrelated in Systor'17_write, MSRC, and FIU. This discrepancy may be associated with highly diverse I/O burstiness in AliCloud, as observed in a prior study [1].

- We perform a Gaussianity test for all four workloads, discovering that I/O burstiness appears to be Gaussian in AliCloud_write and Systor'17_read, but the burstiness is non-Gaussian in AliCloud_read, MSRC, Systor'17_write and FIU. It will be futile if not impossible to truly and accurately describe I/O burstiness in real block storage workload when this distinction is not taken into consideration.
- For cloud storage workloads with a certain degree of correlations, we present visual evidence and theoretical evidence through the autocorrelation function structure of the aggregated process of the request sequences, thereby showing the existence of self-similarity. We further deploy statistical tools to estimate the Hurst parameters. All the estimates are greater than 0.5, confirming the observation above.
- We implement an I/O request series generator, expanding a versatile workload model in which parameters can be obtained directly measured from cloud block traces, to synthesize the temporal and spatial I/O request series for all four sets of block I/O workloads. The experimental results unveil that the proposed generator is conducive to accurately characterizing the burstiness of I/O behaviors.

The rest of this paper is organized as follows. Section II gives an overview of cloud block traces investigated throughout in this work, and the related research studies are summarized in this section. Section III elaborates on the correlation of inter-arrival time of I/O requests in block workloads and performs the Gaussianity test for block workloads. Section IV presents both visual and statistical evidence for the existence of self-similarity in cloud storage workloads with a certain degree of correlations. Section V articulates the implementation of an I/O request series generator aiming to synthesize the temporal and spatial I/O request series for AliCloud, Systor'17, MSRC, and FIU. The significance of block workload characterization is discussed in Section VI. Finally, Section VII concludes this paper.

II. Background and Motivation

A. Cloud Block Storage systems

More often than not, the storage architecture in modern clouds embraces three major components: metadata storage, block storage, and front-end clients. This storage architecture, depicted in Fig. 1, is widely adopted by traditional desktop and server applications in a variety of institutions like Microsoft Research, Fujitsu Laboratory, Florida International University and Alibaba, where intriguing cloud block I/O traces were collected, as summarized in Table I.

MSRC. The MSRC block-level I/O traces, collected more than a decade ago in a data center composed of Microsoft's Windows servers, last for one week and include block-level I/O requests from 179 disks on 13 servers. The overall workload is read-dominated [21], spanning various applications, including

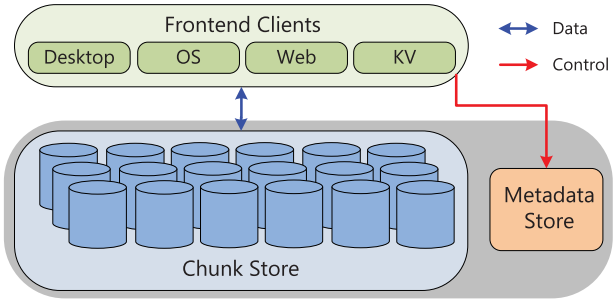


Fig. 1. A cloud block storage system comprises of Chunk Store, Metadata Store and Frontend Clients (application servers).

TABLE I
Summary of MSRC, Systor'17, FIU and AliCloud Traces

Repository:	MSRC [18]	Systor'17 [19]	FIU [20]	AliCloud [1]
Volumes	36	-	-	1,000
Duration	7 days	28 days	21 days	31 days
Reads (Mill.)	304.9	2455.4	55.3	5,058.6
Writes (Mill.)	128.9	898.3	851.1	15,174.4
Read Traffic	9.04 TiB	64.8 TiB	71.19 GB	161.6 TiB
Write Traffic	2.39 TiB	15.0 TiB	642.42 GB	455.5 TiB
Server Categories	file, web, media, etc	VDI	file, web, email	key-value store, web, etc

home directory, web services, media services, and the like. The MSRC traces were extensively analyzed to direct the design and optimization of storage systems, including cloud block storage [2]. Please refer to [18] for the detailed description of the MSRC traces.

Systor'17. To investigate the storage traffic characteristics of virtual desktop infrastructure (VDI), Lee et al. [19] employed several measurement methods, such as a VDI monitoring system, a fibre channel capture system, connection brokers, a system profiler, and usage questionnaire, to glean 28 consecutive days of various types of enterprise storage trace in the commercial office VDI. The overall workload is read-dominated, and a detailed description of Systor'17 trace is articulated in [19].

FIU. To delve into the nature of content similarity and enhance I/O performance, Koller et al. [20] collected I/O traces downstream of an active page cache for a duration of three weeks, on the web, email, and file servers that were used daily by the Department of Computer Science at FIU. The acquired block-level FIU traces, as tabulated in Table I, indicate that FIU workloads are write-dominated. A comprehensive description of the FIU traces can be found in [20].

AliCloud. Unlike MSRC or Systor'17, a cloud environment's write operations dominate I/O workload due to the extensive use of read caches in cloud applications. A production-level cloud block trace, called AliCloud, was collected at a cloud block storage system underlying Alibaba Cloud in 2020. AliCloud includes block-level I/O requests harvested from 1000 volumes and spans various types of cloud applications such as web servers, running operating systems, key-value stores, as summarized in Table I. The detailed description of the AliCloud trace is documented in the literature [1].

To provide insights into cloud block storage system design, storage community often approximately articulate the empirical data of I/O features as traditional distributions such

as normal [11], [13], [14], [15], and it is often assumed that I/O activities follow exponential distribution [16], Poisson distribution [12] and the like, in favor of its attractive theoretical properties and modeling theories. The above approaches, however, do not specialize in lending themselves to accurately characterize the burstiness in block I/O workloads with very bursty I/O activities observed in block storage workloads such as those gleaned from MSRC [21] and AliCloud [1].

In this study, we extract the data sets of both the arrival intervals and the number of read (or write) requests in AliCloud, Systor'17, MSRC, and FIU traces, by merging all the requests in each subtrace (or volume). Taking the Systor'17 case for example, we reorder the timestamps of the read (or write) operations in the trace in chronological order. Then, we calculate the time intervals between two adjacent requests, and the number of read and write requests per second, respectively. Anchored on the aforementioned data sets, we examine the feasibility and effectiveness of using traditional distributions, including normal and Poisson, to delineate I/O arrival behaviors and then analyze the storage system performance. In other words, we address the following fundamental question: for a diversity of cloud applications, is it still appropriate to use independently and identically distributed (IID) methods such as Poisson and normal, to model block I/O activities with intensive burstiness in cloud block storage systems?

B. Related Work

Recently, several works studied the arrival patterns or access patterns in the block-level I/O workloads, such as inter-arrival time of requests [1], [3], [4], numbers of I/O requests [13], reuse time [4], and I/O size [1], [4].

Workload Characterization. From the perspective of data mining, Seo et al. [13] extracted basic properties that can effectively represent I/O workload features, such as the total number of I/O requests and the average interval between I/O requests. After discovering that number of I/O requests is close to normal distribution, Seo et al. devised the clustering algorithms to exploit representative access patterns in the I/O workload. Kashyap et al. [14] analyzed the characteristics of workload on hard drives in enterprise storage systems, and the sample characteristics include transmission length, access pattern, throughput, and utilization. The studies unravel that reads are the primary workload, accounting for 80% of accesses to the hard drive. And write operations presenting burstiness are dominated by short block random accesses. Lee et al. [19] gleaned 28 consecutive days of various types of enterprise storage trace on the commercial office virtual desktop infrastructure. Three observations are: read traffic is dominant, write operations present a local uniform distribution, and the diurnal burstiness is especially obvious. Through analyzing several traces using various distributions, Wajahat et al. [3] developed a stochastic model anchored on a Markov chain to estimate the system performance in storage workloads. Li et al. [1] investigated billions of I/O requests collected from the AliCloud to study the characteristics of load intensity, spatial patterns, and temporal patterns – inter-arrival times of requests. Compared with the well-known public block-level I/O trace, MSRC, the I/O behavior in AliCloud presents more burstiness. This finding may be associated with the trend of increasing load intensity in cloud block storage systems [10].

Self-Similarity. Recently, there were a handful of studies examining self-similarity in different domains, such as cloud workloads [22], Internet traffic [23], and social network dynamics [24]. Using autocorrelation analysis and R/S analysis, Gupta et al. [22] explored Google cluster traces, and the presence of self-similarity and heavy-tailed behavior were detected in cloud workloads. Similarly, Li et al. [23] showed that self-similarity existed in industrial internet traffic. Liu et al. [24] inspected the traces originated from Renren social networks as well as Facebook, and the findings imply that the edge creation process in both networks is consistent with the self-similarity scale.

In the storage community, after gathering spark data for up to six months, Talluri et al. [4] conducted statistical analysis on the popularity of files, read size, inter-arrival time and reuse time: an intriguing conclusion is that read operations showed heavy tail, bursty and negative long-range dependence. For requests in key-value stores, Pitchumani et al. [17] deployed the b-model with only one parameter for self-similar request arrival generation. Nevertheless, it may be unrealistic to accurately model storage workload using only a small set of parameters [25]. Abad et al. [26] investigated six-month traces from two large Hadoop clusters at Yahoo, and the file prevalence, temporal locality, as well as arrival patterns are characterized to resemble the workload conditions. Here is an inspiring discovery: the requests including open, create and delete not only are bursty but also present self-similar behaviors. The aforementioned observations motivated us to delve into the following issue: As I/O activities become a whole lot intensive in modern clouds, does self-similarity exist in representative cloud block storage workloads?

Rooted in four representative block-level cloud workloads, including the read-dominated (MSRC, Systor'17) and write-dominated (AliCloud, FIU) workloads, this work is focused on the appropriateness of capturing intensive I/O behaviors with the traditional distributions. A second focal point of this study is the existence of self-similarity. The third focal point of this work is the synthesis of I/O request sequence.

III. Diagnosing Cloud Block Workloads

To resolve the issues raised in Section II, through correlation study and Gaussianity test, we diagnose AliCloud, MSRC, Systor'17 and FIU to gain a deep understanding of I/O activities in those modern cloud workloads.

A. Correlation Study

In what follows, we make use of the autocorrelation function or *ACF* to study the correlation of inter-arrival times of both read and write requests in the workloads. For a stationary time series $S = \{S_t : t = 1, 2, \ldots, n\}$ with expectation $\theta = E[S_t]$, each time interval – also referred to as $lag - k$ corresponds to an autocorrelation coefficient independent of the time itself.

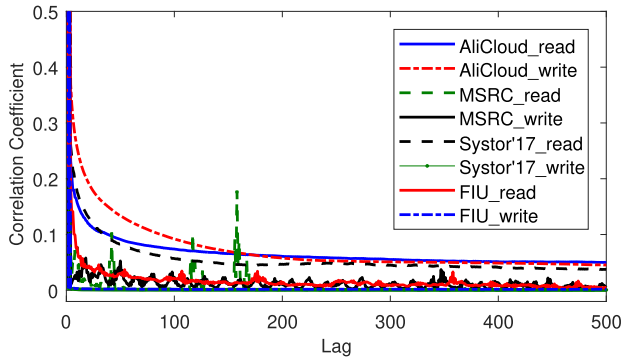


Fig. 2. Auto-correlation functions of inter-arrival time of read and write requests in the AliCloud, MSRC, Systor'17 and FIU workload conditions.

We have $s_t = S_t - \theta$ and then $\{s_t : t = 1, 2, \ldots, n\}$. The autocorrelation coefficients, $ACF(k)$, are calculated at various lags as follows:

$$ACF(k) = \frac{E[s_t \cdot s_{t+k}]}{E[s_t^2]}, \ for\ k \geq 0. \tag{1}$$

where k is the independent variable. A series of autocorrelation coefficients form a functional relationship with various lags, called *autocorrelation function*, which is widely used to measure the interrelationship between the front and back elements in a time series. A detailed introduction of ACF can be found in [27].

The trend of the autocorrelation coefficients is closely related to I/O activities as follows. With an increasing lag, if the correlation coefficient of inter-arrival times of I/O requests rapidly decreases and approaches 0, it entails that there is little correlation for I/O arrivals in storage workload. Then, it may be reasonable to employ an independent and identical distribution to model I/O activities. If the correlation coefficient does not sharply come closer to zero, there is a certain degree of correlation in I/O arrival behaviors.

The autocorrelation functions of inter-arrival times of read and write requests in the AliCloud, MSRC, Systor'17 and FIU workloads are plotted in Fig. 2. The results unveil that there is meager correlation between the arrival intervals for I/O requests in Systor'17_write, MSRC and FIU, especially for I/O requests in FIU, MSRC_write and Systor'17_write. Read requests in MSRC show an unstable weak correlation between the arrival intervals within short-term time scales ($lag < 170$), which may be caused by the fact that MSRC is read-dominated. However, there is insufficient correlation between the arrival intervals for read requests in MSRC in longer time scales. I/O arrival process in this kind of workload may be approximated as independently and identically distributed: the aforementioned Poisson distribution tends to be a superb modeling tool.

Unlike Systor'17_write, MSRC and FIU, AliCloud and Systor'17_read workloads exhibit a stable correlation between the arrival intervals for requests within a long-term time scale ($lag = 500$). For AliCloud, this phenomenon is aligned with a recent finding reported in the literature – the I/O features of the various types of upper-level cloud applications managed by the AliCloud block storage system are often quite different [1], indicating that independently and identically distributed time series are not suitable to characterize I/O activities in AliCloud. Furthermore, the existence of the long-term correlation of inter-arrival times of I/O requests in AliCloud and Systor'17_read inspires us to further explore the self-similarity in cloud storage workloads.

TABLE II
SUMMARY OF THE DIAGNOSES FOR ALICLOUD, MSRC, SYSTOR'17 AND FIU

Test	AliCloud		MSRC		Systor'17		FIU	
	read	write	read	write	read	write	read	write
ACF	√	√	×	×	√	×	×	×
Gauss	×	√	×	×	√	×	×	×

Below we summarize two key observations:

Observation 1: There is almost no correlation between request arrival intervals in the MSRC, FIU and Systor'17_write workloads.

Observation 2: There is a certain degree of correlation in the long-term timescale for request arrival intervals in the AliCloud and Systor'17_read workloads.

B. Gaussianity Test

Previous studies indicate that Gaussianity or non-Gaussianity is an important factor that must be carefully considered in constructing models for performance optimization [11], [28]. Gaussian test facilitates the accurate description of tail trends in the distribution of I/O characteristics, thus forging more accurate models. Therefore, we opt for the deployment of Gaussianity test to investigate cloud workloads in this section.

Gaussianity test (Gauss) can be performed through quantile-quantile (QQ) plots. The quantile of a random variable, $X = \{X_t : t = 1,\ 2, \ldots\}$, refers to the real number x that satisfies the condition $P(X_t \leq x) = c$, where c is a constant. The QQ plot is the trajectory formed by the points (x, y) corresponding to the quantiles x, y for two random variables X and Y. If these two data sets share the same distribution, points (x, y) will fall on a line with a 45-degree angle, and vice versa.

Fig. 3 depicts the QQ plots of block trace data versus standard normal with respect to I/O requests in AliCloud, MSRC, Systor'17 and FIU. Fig. 3(b) and (e) show that the offset of the scatter curve of AliCloud_write and Systor'17_read is relatively small, the major scatters of AliCloud_write and Systor'17_read fall approximately on a straight line, indicating that both read requests in Systor'17 and write requests in AliCloud appear to be Gaussian. The other plots in Fig. 3 show that the scatters of AliCloud_read, MSRC, Systor'17_write, and FIU obviously do not fall on a straight line, and the entire curves are concave upward, suggesting that I/O requests in AliCloud_read, MSRC, Systor'17_write, and FIU appear to be non-Gaussian.

Table II summarizes the aforementioned diagnoses for AliCloud, MSRC, Systor'17 and FIU. If the corresponding request activity shows the Gaussianity or a certain degree of correlation, a tick is used; otherwise, a cross is used. Table II confirms the diversity of workload characteristics. In a given workload, the arrival process can be autocorrelated and but not Gaussian, or vice versa. For example, even though there is a certain degree of correlation in AliCloud_read and Systor'17_read,

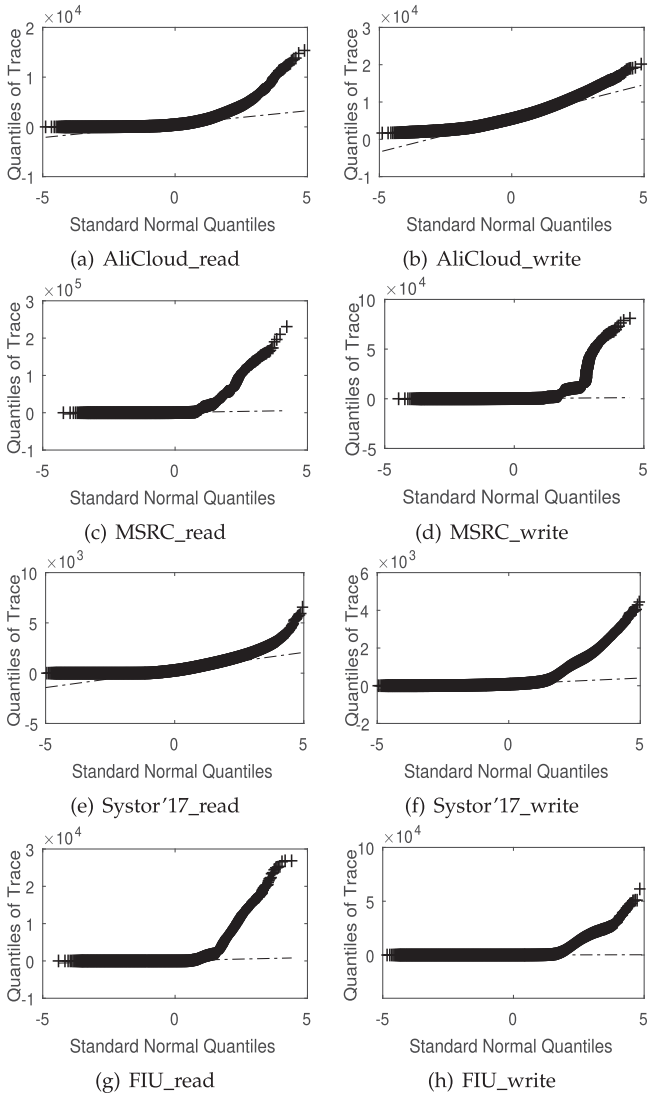


Fig. 3. Examine the Gaussianity of I/O requests in AliCloud, MSRC, Systor'17 and FIU through QQ plots of block trace data versus standard normal, respectively.

Systor'17_read is Gaussian, but AliCloud_read is obviously non-Gaussian.

The following summarizes our major observations:

Observation 3: Read requests in Systor'17 and write requests in AliCloud appear to be Gaussian.

Observation 4: Requests in AliCloud_read, MSRC, FIU and Systor'17_write tend to be non-Gaussian.

Observation 5: Even though AliCloud_read and Systor'17_read appears to show a certain degree of correlation, Systor'17_read is Gaussian but AliCloud_read is obviously non-Gaussian.

IV. Self-Similarity Analysis

Since the AliCloud and Systor'17_read workloads embrace a certain degree of correlation, in this section we deploy the mature theory and statistical techniques to explore the existence of self-similarity in these block workloads through three approaches: (1) visual evidence (see Section IV-A which takes AliCloud for example), (2) the autocorrelation structure of the aggregated process of request sequences (see Section IV-B), and (3) Hurst parameter measurement (see Section IV-C).

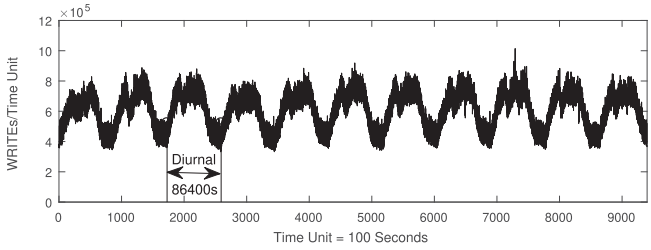


Fig. 4. Diurnal rhythm of write requests (WRITEs) in AliCloud. .

A. Visual Evidence

We observe that in the AliCloud workload I/O activities present clearly diurnal rhythm. For example, the peak and trough of write requests in AliCloud, as well as the changing trend in each day, are almost identical in each period, as evidenced in Fig. 4. This finding is unsurprising, and it is consistent with the temporal properties exhibited in disk, network and web traffics [17]. On the other hand, the prior studies on LAN traffic suggest that data traffic gradually loses self-similar feature after exceeding the time scale of day [29]. Therefore, it is viable for us to adopt data on several timescales within a day to intuitively present the self-similarity in AliCloud.

Fig. 5 unfolds both read and write request arrival rates in AliCloud at three different time scales. The horizontal axis represents the time scale, and the vertical axis represents the number of requests per unit of time. Each plot is a subinterval randomly extracted from the time range of the latter plot, and the time resolution is increased by 10 times. For example, plots (a)–(c) shows read request arrival rates. Plot (a) illustrates a randomly drawn period (100 seconds) drawn from plot (b), and plot (b) depicts a short period (1000 seconds) randomly derived from plot (c). The same pattern is applied to plots (a')–(c'), which demonstrates write request arrival rates.

In sum up, the time range in which I/Os are bursty consists of a raft of subintervals that have similar burst behaviors. Furthermore, each of such subintervals is made of even smaller subintervals with similar burst behaviors. As result, I/O activities in AliCloud appear to be self-similar over the long-term time scales.

Observation 6: I/O activities in AliCloud present clearly diurnal rhythm.

Observation 7: I/O activities in AliCloud appear to be self-similar.

B. Theoretical Evidence

A covariance stationary stochastic process $X = \{X_t : t = 1, 2, 3, \ldots\}$, $X^{(m)} = \{X_t^{(m)} : t = 1, 2, 3, \ldots\}$ is called as the corresponding m-order aggregated process, if expression

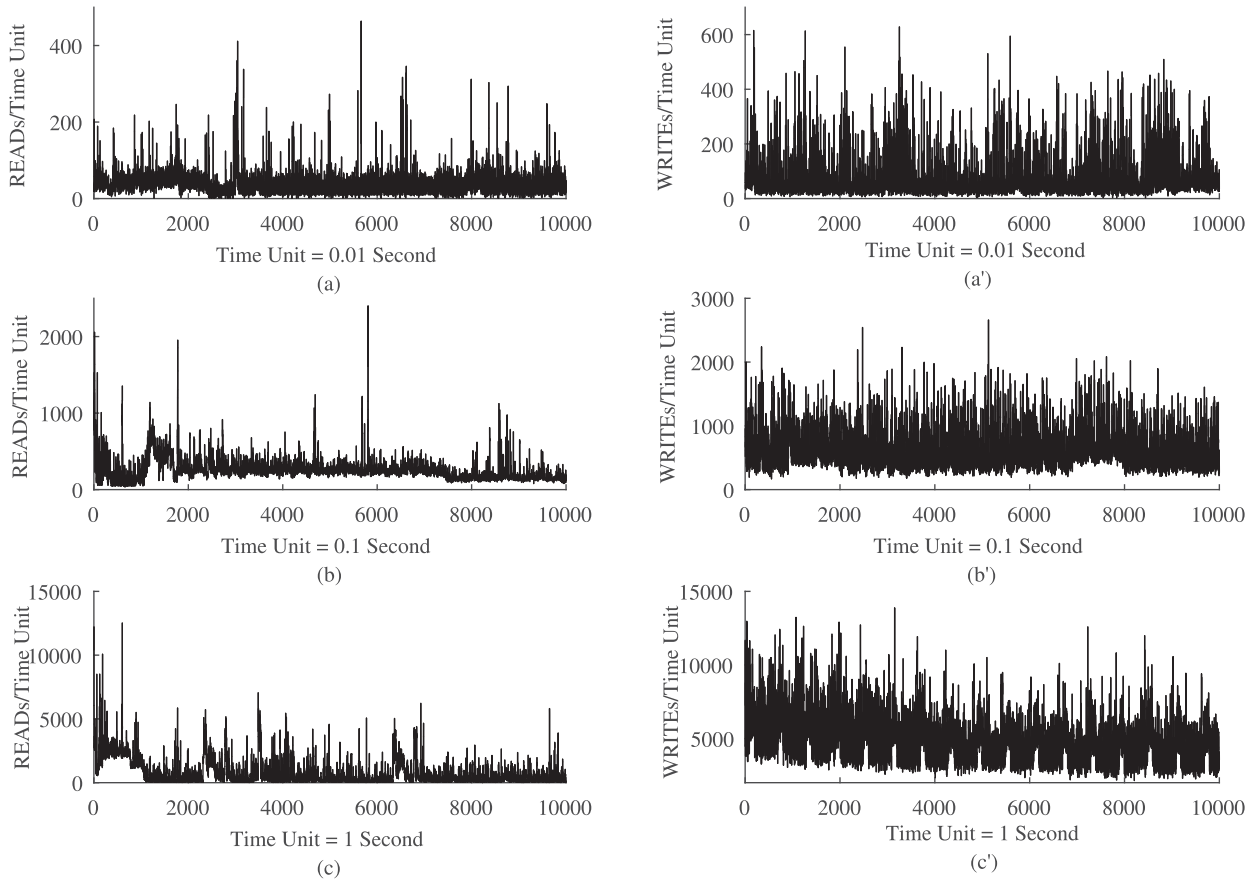


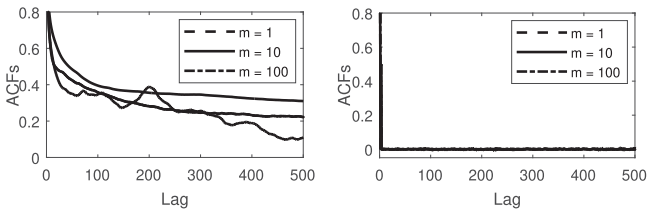
Fig. 5. Visualization of AliCloud block trace: Left plots illustrate the number of read requests (READs) per time unit for three different time scales (a)–(c), and right plots present the number of write requests (WRITEs) per time unit for the same time scales (a′)–(c′). Each plot has ten thousand buckets.

$X_t^{(m)} = \frac{1}{m}\sum_{i=0}^{m-1} X_{tm-i}$ is held. Hence, we denote the corresponding ACF as $ACF^{(m)}(k)$. For the traditional stochastic process, such as Poisson, $ACF^{(m)}(k)$ *degenerates* with the increase of m, and converges to 0 (i.e., $ACF^{(m)}(k) \to 0,\ as\ m \to \infty$). If the structure of ACF for $X^{(m)}$ tends to be the same autocorrelation function structure rather than *degenerating* with an increasing m, i.e., as $m \to \infty$

$$ACF^{(m)}(k) \to \frac{1}{2}\left[(k+1)^{2-\lambda} - 2\,k^{2-\lambda} + (k-1)^{2-\lambda}\right], \quad (2)$$

the process X is said to be self-similar with the *Hurst parameter* H. The Hurst parameter is the sole parameter gauging the degree of self-similarity. A Hurst parameter in the range of $(0.5, 1)$, indicating the presence of self-similarity which is also called *long-range dependence (LRD)*. This statement constitutes a theoretical basis for exploring the existence of self-similarity, and it entails that self-similar workloads exhibit burstiness across different time scales, due to the scale-invariant property that is similar from different time scales.

When it comes to the Systor'17_read case, we plot the autocorrelation functions of the aggregated time series of read request series at multiple aggregation levels in Fig. 6(a) where the x-axis represents the number of requests [30], and we draw the autocorrelation functions of the corresponding artificial Poisson workload at each aggregation level in Fig. 6(b). The aggregation level (m) in Fig. 6 is 1, 10, and 100, respectively.

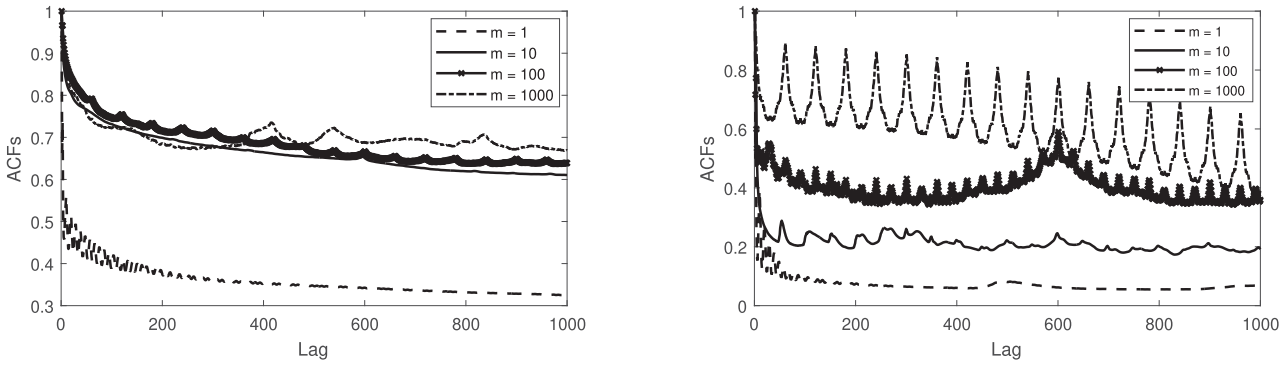


(a) Systor'17_read (b) Poisson workload

Fig. 6. Autocorrelation functions of the aggregated time series for (a) Systor'17_read, and (b) artificial Poisson workload.

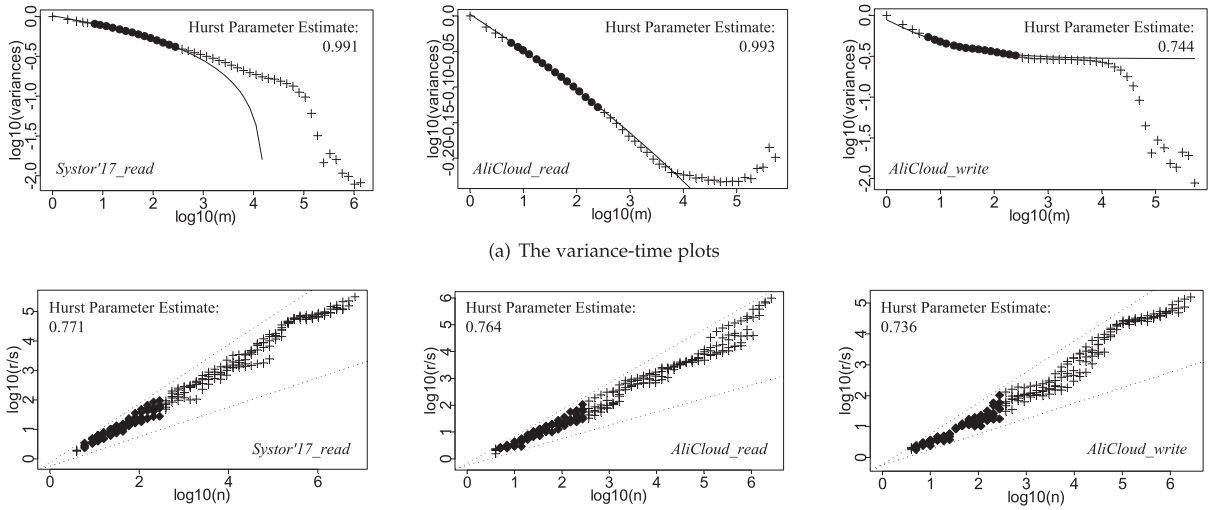
For Systor'17_read requests shown in Fig. 6(a), the autocorrelation coefficients of the aggregated time series obviously do not converge to 0 as the lag increases from 1 to 500. The corresponding curves approach the same function structure. In contrast, as illustrated in Fig. 6(b), the autocorrelation coefficients of artificial Poisson workload at all the aggregation levels are nearly 0. The above observations unfolds that the autocorrelation structure of read request series for Systor'17 is completely dissimilar from that of the Poisson workload.

By the same token, for read and write requests in AliCloud, we further plot the autocorrelation functions at different aggregation levels (m is 1, 10, 100, and 1000) in Fig. 7(a) and (b), respectively. The autocorrelation coefficients of the aggregated time series of read and write request series in AliCloud also do



(a) AliCloud_read (b) AliCloud_write

Fig. 7. Autocorrelation functions of the aggregated time series for (a) AliCloud_read, and (b) AliCloud_write.



(a) The variance-time plots

(b) The Pox plots

Fig. 8. Estimating Hurst parameter for request series in Systor'17_read and AliCloud: (a) Variance-time plots; and (b) Pox plots.

not converge to 0 as the lag rises from 1 to 1000. In addition, the corresponding curves in each plot exhibit regular fluctuations and appear to have the same function structure. This finding is overlapping with that observed in the Systor'17_read request series.

In summary, request activities in both AliCloud and Systor'17_read behave similarly to a self-similar process.

Observation 8: The autocorrelation structure of request series at different aggregation levels indicates that request activities in both AliCloud and Systor'17_read behave similarly to a self-similar process.

C. Measurement of Hurst Parameter

We advocate for two popular statistical analysis tools, variance-time plot [31] and R/S (rescaled adjusted range) analysis [32] (also called Pox plot), to estimate the Hurst parameters for requests issued in Systor'17_read and AliCloud.

Fig. 8(a) unveils the variance-time plots of request series in Systor'17_read and AliCloud. Fig. 8(a) draws the curve of $log_{10}(Var(X^{(m)}))$ (briefly noted as $log_{10}(variances)$) versus $log_{10}(m)$, and the corresponding Hurst parameter is estimated to be 0.991, 0.993 and 0.744, respectively. In addition, Fig. 8(b) presents the Pox plots resulting from the R/S analysis of request series in Systor'17_read and AliCloud block trace data. Using the least squares linear fitting, we obtain the Hurst parameter as 0.771, 0.764, and 0.736.

The Hurst parameter estimates for request series in Systor'17_read and AliCloud are significantly greater than 0.5, which quantitatively stipulates the existence of self-similarity.

Observation 9: All the Hurst parameter estimates are greater than 0.5 in Systor'17_read and AliCloud, thereby confirming the existence of self-similarity.

V. Synthesizing Request Sequences

For the cloud storage community, synthetic models are slated to offer a deep understanding of the characteristics of offline workloads, including I/O bursty activities, heavy-tail property, and the like. As I/O request activities involve both temporal and

spatial locality, this section is dedicated to a versatile model to accurately capture the temporal and spatial distribution of block I/O requests.

A. The Alpha-Stable Model

By plugging in the correlation study, Gaussianity test, and self-similarity analysis on these four representative cloud block storage workloads, we record the following complex observations: (1) I/O bursty generally exists in the read-dominated and write-dominated cloud block workloads, (2) I/O activities in some block workloads, including Systor'17_write, MSRC, and FIU, appear to be independently and identically distributed, whereas AliCloud and Systor'17_read evidently display self-similar, and (3) I/O activities resemble the Gaussian condition in Systor'17_read, and AliCloud_write, and the non-Gaussian one in MSRC, AliCloud_read, Systor'17_write, and FIU.

To effectively synthesize these cloud block workloads containing complex I/O activities, we advocate for employing a versatile trace generator to not only accurately synthesize both the IID and self-similar workloads but also flexibly characterize burstiness under both the Gaussian and non-Gaussian conditions for block workloads. This feature is a shinning example of the strengths embraced by the alpha-stable process: the observation galvanizes us to extend the alpha-stable workload model reported in the literature [33] to devise the temporal and spatial I/O request series in cloud block workloads for AliCloud, Systor'17, MSRC, and FIU, respectively.

A random variable X that follows an alpha-stable distribution is denoted by $X \sim S^{\alpha}_{\sigma,\beta,\mu}$ [33], and the formalization of the alpha-stable workload model can be expressed as follows:

$$REQUEST(i) = v \cdot N_{\alpha,\beta,H}(i) + \delta, \tag{3}$$

where $REQUEST(i)$ represents the number of requests in the i^{th} unit time or the i^{th} block. Similarly, we establish the following parameters from the temporal and spatial perspectives, respectively: α measures the degree of burstiness, β represents the degree of the heavy tail in block workloads, and variables σ and μ measure the scale and location parameters. In addition, variable v is referred to as the mean number of requests per unit time (or per block) in block workload, H denotes the degree of self-similarity, δ gauges the standard deviation of the number of requests per unit time (or per block) relative to the mean. The detailed description of $N_{\alpha,\beta,H}(i)$ can be located in the literature [33].

The values of all the aforementioned parameters are estimated and derived from the real cloud block traces. In this study, the maximum likelihood estimation is used to measure I/O request series in the AliCloud, Systor'17, MSRC, and FIU trace samples to estimate the parameters used in the alpha-stable request series generator.

When it comes to the read/write-request datasets articulated in Section II-A, there tend to be orders of magnitude discrepancies in dataset sizes for various workloads. Given AliCloud, Systor'17, and FIU_write workloads with particularly large data volumes, we take the first one million data items as a temporal sample sequence. The projected values are tabulated in Table III.

TABLE III
ESTIMATED PARAMETERS OF ALPHA-STABLE MODEL BASED ON THE MAXIMUM-LIKELIHOOD ESTIMATE FOR THE TEMPORAL SAMPLE SEQUENCES

Trace name	I/O type	Alpha-stable parameter				v	δ
		α	β	σ	μ		
AliCloud	read	1.488	1.000	296.8	262.7	710.3	803.9
	write	1.900	1.000	1251	5288	5750	1886
Systor'17	read	2.000	0.000	273.1	337.0	433.0	389.8
	write	1.038	0.9286	34.28	77.00	151.6	202.2
MSRC	read	0.5481	0.6839	283.1	69.26	5985	16073
	write	0.7760	0.7615	76.70	153.1	806.8	3039
FIU	read	0.5115	1.000	3.605	-48.85	585.9	1814
	write	0.5327	1.000	3.347	4.658	336.6	1734

TABLE IV
ESTIMATED PARAMETERS OF ALPHA-STABLE MODEL BASED ON THE MAXIMUM-LIKELIHOOD ESTIMATE FOR THE SPATIAL SAMPLE SEQUENCES

Trace name	Alpha-stable parameter				v	δ
	α	β	σ	μ		
AliCloud	1.1708	1.000	0.24957	0.63755	4.8613	256.78
Systor'17	0.9358	1.000	0.69412	1.1642	6.8534	283.31
MSRC	0.9658	1.000	1.9350	0.21095	33.766	2017.6
FIU	0.9909	1.000	0.18884	0.64576	1.9688	1.9312

By the same token, we calculate – in the context of the spatial I/O request series – the number of I/O requests on each block for AliCloud, Systor'17, MSRC and FIU workloads. Due to space limitation, we target the I/O request datasets corresponding to DiskNumber 0 in MSRC, LUN 0 in Systor'17, block major number 2 in FIU, and volume 0 in AliCloud, thereby sampling the first one million data as the spatial sample sequences. The estimated values are tabulated in Table IV.

Since the alpha-stable stochastic process will degenerate into a Gaussian stochastic process as α is closing to 2 [33], and the estimate of α for AliCloud_write and Systor'17_read in Table III are 1.9 and 2, meaning that write request series in AliCloud and read request series in Systor'17 appear to be Gaussian – a trend that is fully consistent with the test results given in Section III and supports the Gaussianity test results.

B. Temporal Analysis

Anchored on the parameter estimated in Table III, we apply the alpha-stable workload model to generate synthetic requests, and we compare the synthesized request sequences to the real sequences, aiming to analyze the accuracy of our model. To convey confidence to the accuracy of the proposed model, we adopt the trimmed mean of errors, empirical distribution, and relative error to assess the bias between each real request sequence and the synthetic series.

1) Trimmed Mean of Errors: The trimmed mean of data set is the arithmetic mean after cutting off a smaller proportion of data at both ends of the sample data, which is more stable to abnormal data than the usual sample average, such as the arithmetic mean.

We implement the alpha-stable model – the PatIO generation algorithms based on the Pareto distribution [34] – accompanied by the typical self-similar workload models, including fractional Brownian motion (FBM) [35] and fractional auto-regressive integrated moving average (FARIMA) [36], to generate I/O

TABLE V
THE TRIMMED MEANS OF ERRORS FOR SYNTHESIZING THE TEMPORAL REQUEST SEQUENCES

Trace name	I/O type	Self-similar method		IID method		PatIO ⑤	Proposed	Improvement				
		FBM ①	FARIMA ②	Normal ③	Poisson ④			vs ①	vs ②	vs ③	vs ④	vs ⑤
AliCloud	read	1913.7	473.39	-	-	5618.8	437.81	77%	8%	-	-	92%
	write	5257.5	5554.6	-	-	23587	1777.2	66%	68%	-	-	92%
Systor'17	read	937.91	352.95	-	-	5824.1	365.97	61%	-4%	-	-	94%
	write	-	-	163.45	76.969	869.83	72.833	-	-	55%	5%	92%
MSRC	read	-	-	13027	5873.9	1892.2	2000.2	-	-	85%	66%	-6%
	write	-	-	641.11	640.83	493.31	234.31	-	-	63%	63%	53%
FIU	read	-	-	1407.0	578.29	132.18	105.56	-	-	92%	82%	20%
	write	-	-	1270.0	319.92	40.578	29.832	-	-	98%	91%	26%

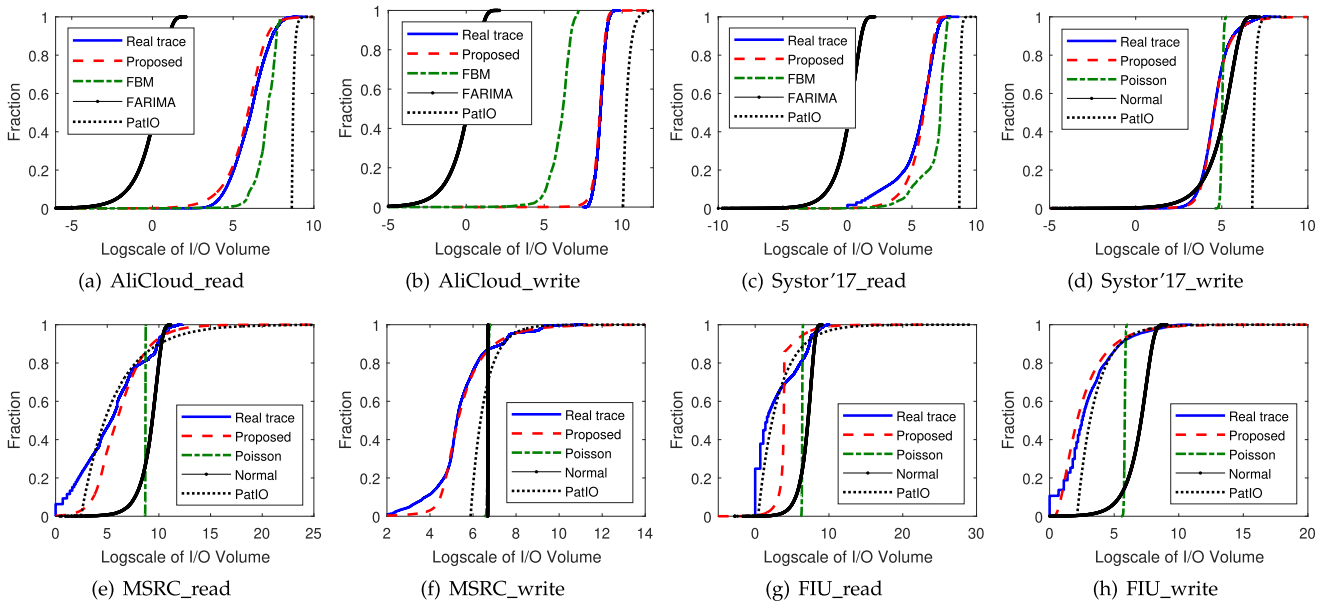


Fig. 9. Comparison of CDFs between synthetic trace and real trace for the temporal request sequences, respectively.

sequences for AliCloud and Systor'17_read. The corresponding trimmed mean of errors are recapped in Table V.

More often than not, the trimmed means of error between the real request sequences and the alpha-stable synthetic series are minimal in the AliCloud cases. Moreover, there are different degrees of improvement compared with the typical methods, and the improvement degrees of the trimmed means of error for write request are even more than 66%. Even for the Systor'17_read case, the trimmed mean of error between the real workload and the alpha-stable synthetic workload is also close to the minimum. The trimmed mean of errors for the alpha-stable synthetic workload and the minimum error are only a 4% gap, indicating that the matching degree between the genuine workload and the alpha-stable synthetic workload is still reasonably good.

We also employ the alpha-stable generator and the conventional normal, Poisson, and PatIO methods to generate I/O sequences for the independently and identically distributed Systor'17_write, MSRC, and FIU workloads. The corresponding trimmed mean of errors are also summarized in Table V. Generally speaking, except the PatIO synthetic series for MSRC_read, the trimmed means of error between the real request sequences and the alpha-stable synthetic sequences are relatively small for the I/O request series in Systor'17_write, MSRC_write, and FIU. When it comes to the MSRC_read case, the difference between the trimmed mean of errors for the alpha-stable synthetic workload and the minimum error is merely 6%, unfolding that the matching degree between the genuine workload and the alpha-stable synthetic workload is reasonably good. Moreover, there are different degrees of improvement compared against the traditional Poisson and normal methods, and the improvement rate of the trimmed means of error in the Systor'17_write, MSRC and FIU cases are almost consistently exceeding 55%.

2) An Empirical Study: In order to intuitively contrast the matching degrees between the various synthetic series and real series, we plot the cumulative distribution functions (CDF) of the real and synthetic series for AliCloud, Systor'17, MSRC and FIU in Fig. 9. Fig. 9(a)–(c) show the matching degree of the self-similar request series in the AliCloud and Systor'17_read scenarios, and Fig. 9(d)–(h) plot the matching degree of the independently and identically distributed request series in the Systor'17_write, MSRC and FIU cases, respectively.

The horizontal axis in Fig. 9 represents the logscale of I/O requests per unit time, and the vertical axis denotes the proportion of the corresponding values in the whole request series. A point

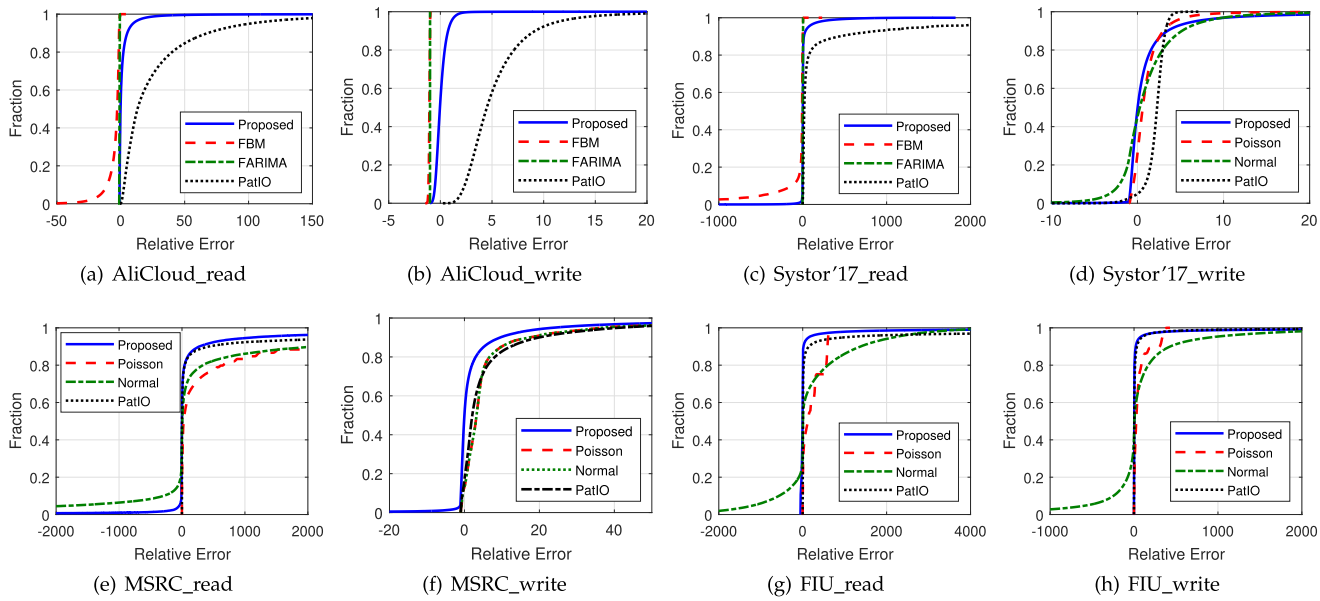


Fig. 10. Comparison of relative errors between synthetic trace and real trace for the temporal request sequences, respectively.

(x, y) in the distribution curve indicates that the proportion of I/O volume less than or equal to x in the corresponding request series is y.

Fig. 9 reveals that the request sequences synthesized by the proposed model well match the real trace, especially for the self-similar series in AliCloud and Systor'17_read, and the IID request series in Systor'17_write, MSRC_write and FIU_write. The corresponding trimmed means of errors are 437.81, 1777.2, 365.97, 72.833, 234.31 and 29.832, respectively, as shown in Table V. These trimmed means of errors for the AliCloud and Systor'17_read cases are somewhat large, which is partially due to the high density of requests in AliCloud and the read-dominated Systor'17. With respect to AliCloud, for example, the trimmed means of errors – 437.81 and 1777.2 – are small relative to the thousands of requests per second.

Although the trimmed mean of errors between the real sequence and the alpha-stable synthetic sequence from the perspective of Systor'17_read is greater than the FARIMA synthetic sequence (see Table V), Fig. 9(c) unveils that the alpha-stable synthetic sequence can more accurately characterize the request behaviors in the actual workload. This observation confirms that the accuracy of the proposed synthetic model is credible for the Systor'17_read case.

Furthermore, except for the PatIO synthetic sequences in Fig. 9(e) and (g), there are different degrees of improvement relative to all of the other synthetic sequences. For instance, the alpha-stable synthetic series reduce the trimmed mean of error of the PatIO synthetic series in the AliCloud, Systor'17, MSRC_write and FIU_write scenarios by 92%, 92%, 94%, 92%, 53%, and 26% (see Table V), respectively.

3) Relative Error: To make the accuracy of the proposed model more trustworthy, we further adopt the relative error – an evaluation metric widely accepted by the storage community [13], [25] – to test the matching degrees between various synthetic series and real series.

Fig. 10 depicts the empirical distribution of the relative errors between the various synthetic series and real series in the context of AliCloud, Systor'17, MSRC, and FIU. Fig. 10(a)–(c) show the relative errors of the self-similar request series in the AliCloud and Systor'17_read scenarios, and Fig. 10(d)–(h) plot the relative errors of the independently and identically distributed request series in the Systor'17_write, MSRC, and FIU scenarios, respectively.

The horizontal axis of Fig. 10 represents the relative error, whereas the vertical axis denotes the proportion of the corresponding values in the whole relative errors. A point (x, y) in the distribution curve indicates that the proportion of relative error is less than or equal to x in the corresponding request series is y.

The higher the proportion of values with a relative error close to 0, the better the matching degree between the synthetic series and real series is. Fig. 10 reveals that the matching degree between the proposed synthetic series and real series is the best for almost all the sample sequences.

Although the matching degrees between the real sequences and the alpha-stable synthetic sequences from the perspectives of MSRC_read and FIU_read are not sufficiently satisfactory in Fig. 9, Fig. 10(e) and (g) unravel that compared to the PatIO synthetic sequences, the relative errors of the alpha-stable synthetic sequences have an edge in capturing the request characteristics of the actual MSRC_read and FIU_read sequences.

In summary, for the independently and identically distributed and self-similar block workloads, or for the Gaussian and non-Gaussian workloads, or for the read-dominated and write-dominated workloads, the matching degree between the real trace series and the alpha-stable synthetic sequence is encouraging, satisfying, and practical.

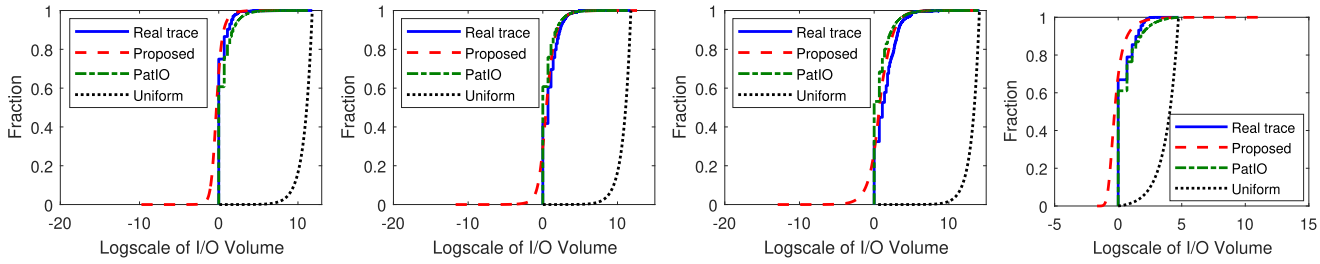


Fig. 11. Comparison of CDFs between synthetic trace and real trace for the spatial I/O request sequences, respectively.

TABLE VI
THE TRIMMED MEANS OF ERRORS FOR SYNTHESIZING THE SPATIAL I/O SERIES

Trace name	PatIO ⑤	Uniform ⑥	Proposed	Improvement	
				vs ⑤	vs ⑥
AliCloud	0.6811	66971	0.5738	16%	99.9%
Systor'17	1.7765	65956	2.0107	-13%	99.9%
MSRC	4.1258	66152	5.6686	-37%	99.9%
FIU	0.8647	56.072	0.6858	21%	98.8%

C. Spatial Analysis

Using the parameters speculated in Table IV, we apply the alpha-stable workload model to originate spatial requests. We also employ the aforementioned three tools elaborated in Section V-B to gauge the corresponding errors.

1) Trimmed Mean of Errors: We implement the alpha-stable model, the PatIO generation algorithms furnished by the Zipf distribution [34], and the typical uniform method, to forge spatial I/O sequences catering for AliCloud, MSRC, Systor'17 and FIU. We compare the synthesized request sequences to the authentic sequences and tabulate the corresponding trimmed mean of errors in Table VI.

Glancing at Table VI, the trimmed means of errors of the synthetic sequences for the proposed model seem to be non-minimum in the Systor'17 and MSRC scenarios. Although the trimmed means of errors between the real sequences and the alpha-stable synthetic sequences from the perspectives of Systor'17 and MSRC are slightly larger than that of the PatIO synthetic sequences, the margin of 0.2 and 1.5 over the corresponding minimum errors can be ignored. The results indicate that the matching degree between the real workload and the alpha-stable synthetic workload is satisfactory.

Another intriguing finding observed in Table VI is that the trimmed means of error of the synthetic sequences for the proposed and PatIO methods are much smaller than the uniform synthetic sequences, and the improvement rates of the trimmed means of error for all the cases are almost consistently above 99%.

2) An Empirical Study: To intuitively contrast the matching degrees between the various synthetic series and real series, we plot the cumulative distribution functions of the authentic and synthetic series for AliCloud, Systor'17, MSRC, and FIU in Fig. 11.

In Fig. 11, the horizontal axis represents the log scale of I/O requests per block, and the vertical axis denotes the proportion of the corresponding values in the entire request series. A point (x, y) in the distribution curve implies that the proportion of I/O volume less than or equal to x in the corresponding request series is y.

Fig. 11 reveals that the spatial request sequences synthesized by the proposed model are well aligned with the real traces, especially for the I/O request series in the AliCloud, Systor'17, and MSRC cases. Thus, although the corresponding trimmed means of errors for Systor'17 and MSRC – 2.0107 and 5.6686 listed in Table VI – are non-minimum, the alpha-stable synthetic sequences are still convincing. This finding once again demonstrates that the accuracy of the proposed synthetic model for the spatial scenarios is no less than PatIO.

3) Relative Error: To incorporate confidence to the accuracy of the alpha-stable workload model, we further adopt the relative error metric to measure the matching degrees between the various synthetic series and real series. Fig. 12 depicts the empirical distribution of the relative errors between the various synthetic series and real series of AliCloud, Systor'17, MSRC, and FIU, respectively.

The horizontal and vertical axes in Fig. 12 denote the relative error and the proportion of the corresponding values in the whole error series. A point (x, y) in the distribution curve suggests that the proportion of relative error less than or equal to x in the corresponding error series is y. A high proportion of values with a relative error close to 0 attests a prominent matching degree between the synthetic series and authentic series.

Fig. 12 reveals that for the proposed synthetic request sequences, the proportions of the relative error close to 0 for all of the cases are almost consistently above 80%, and the measures of AliCloud and FIU impressively exceed 95%. More interestingly, the empirical distribution of the relative errors for the proposed synthetic sequences is almost identical to that for the PatIO synthetic sequences. These observations illustrate that both the proposed and PatIO synthetic sequences satisfyingly mirror the spatial sequences in the AliCloud, Systor'17, MSRC, and FIU scenarios.

The key takeaway from the perspectives of the trimmed mean of errors, empirical distribution, and relative error is that for the synthesis of the spatial I/O series, the accuracy of the alpha-stable model is no worse than the PatIO method. We conclude that the matching degree between the genuine trace series and the alpha-stable synthetic sequence is satisfactory and desirable.

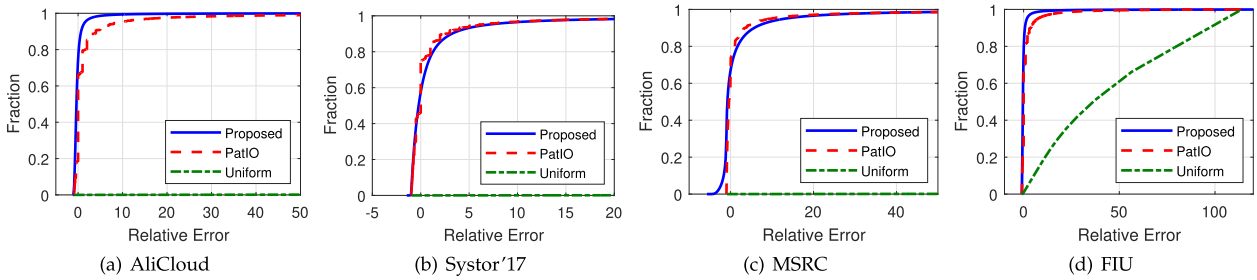


Fig. 12. Comparison of relative errors between synthetic trace and real trace for the spatial I/O request sequences, respectively.

VI. Discussions

Block workload Characterization is often used to design and optimize data storage stacks. In this section, we discuss the significance of characterizing cloud block workload.

A. Block Workload Characteristics

Oftentimes the design and optimization of cloud block storage systems rely on insights derived from studying block-level storage workloads. For example, through profiling applications where I/O traces are similar to one other in their I/O behaviors, Kang et al. [37] proposed a framework that effectively estimates the execution time of I/O traces on solid-state drives (SSDs), thereby fostering both users and manufacturers to accurately evaluate SSD. Based on the characteristics of cloud storage workload, Mao et al. [38] proposed a hybrid data distribution method that distributes data blocks with replication or erasure code schemes, to judiciously and adaptively distribute data blocks among multiple cloud storage providers. Yadgar et al. [39] found that specific I/O workload features like logical locality strongly affect write amplification, read amplification and flash read costs, thus providing opportunities for optimizing SSD design.

Recently, cloud providers like the Alibaba cloud, widely employed hybrid storage nodes consisting of SSDs and hard-drive disks (HDDs). These hybrid storage nodes are usually in SSD write-back (SWB) mode, writing incoming data to their SSDs, followed by refreshing the data to the corresponding hard disks to ensure a lower write latency. The characteristics of real production workload from Pangu, a large-scale storage platform underlying AliCloud, unveil that a plethora of write-dominated storage nodes (WSNs) exist, and the SSDs of these WSNs have severe high-write intensity and long tail latency in the SWB mode. Then, Liu et al. [40] proposed a run-time I/O scheduling mechanism for WSNs, the SSD write redirect, to fully alleviate the above problems and significantly improve the overall system performance and SSD durability. Wang et al. [41] proposed a buffer-controlled writing method to actively control buffer writing, and a hybrid I/O scheduler was devised to adaptively direct incoming data to SSDs and HDDs to minimize write latency.

B. Distribution of I/O Characteristics

To accurately understand block-level I/O workload characteristics, one ought to master the distribution of I/O characteristics to effectively predict the system performance in storage workload by the virtue of an accurate stochastic model according to the corresponding distribution. For example, through investigating AliCloud's empirical distributions of I/O patterns, including inter-arrival times of requests, Li et al. [1] unfolded 15 new discoveries, which furnish optimization insights into load balancing, cache efficiency, and storage cluster management in cloud block storage systems. Wajahat et al. [3] performed a distribution fit to I/O features (e.g., inter-arrival times) in the different workloads and developed a stochastic model based on Markov chain to evaluate system performance in storage workloads. After studying the empirical distribution of various features in the block-level I/O workloads of common applications on nexus5 smartphones, Zhou et al. [42] proposed an array of ideas including fast serving small requests to enhance the overall performance of smartphone storage subsystems.

In recent years, with the escalated load intensity in cloud workloads, I/O behaviors at the block level become increasingly more bursty - a challenge for accurately modeling I/O patterns. In this work, four representative sets of block I/O traces are placed under the microscope to diagnose the correlation of inter-arrival times of request series in block workload, and Gaussianity, diurnal rhythm, burstiness, and self-similarity in I/O patterns. Then, an I/O sequence generator based on the alpha-stable model is proposed to synthesize workloads matching the real I/O property. Such a workload generation method is highly configurable, and it is slated to be deployed by modifying model parameters to offer good scalability. In addition, the proposed model embraces good robustness measure, thereby being applied to generate I/O request series in multiple representative cloud block workloads such as those used throughout this study.

Synthetic workloads can be replayed on a real cloud block storage system to simulate I/O behaviors. Through generating block I/O workload, we expect to help system administrators in evaluating and optimizing I/O performance by gaining workload characteristics like system throughput. Therefore, our full-fledged modeling solution becomes a vital and pragmatic auxiliary means of system performance evaluation.

VII. Conclusion

With an accurate understanding of target workloads, it is viable to revamp the performance of cloud block storage systems running various applications, including deduplication, caching, metadata management, replication, energy saving. This work

furnishes a novel method for understanding and accurately characterizing I/O behaviors in cloud block storage workloads.

In this study, we analyzed four representative sets of block I/O traces, namely, the AliCloud, Systor'17, MSRC, and FIU workloads, gleaned in the classic application environments. After delving into the correlation of request arrival intervals in block I/O workloads, we observed that there is an unequivocal correlation in the long-term timescale for request arrival intervals in AliCloud and Systor'17_read. In contrast, when it comes to Systor'17_write, MSRC and FIU, request arrival intervals are almost uncorrelated. This discrepancy may be associated with the more diverse I/O burstiness in AliCloud.

We also performed the Gaussianity test for all the four traces, and we discovered that I/O burstiness appears to be Gaussian in AliCloud_write and Systor'17_read, whereas it is likely to be non-Gaussian in AliCloud_read, MSRC, Systor'17_write, and FIU. It will be arduous to genuinely and accurately capture I/O burstiness in real block storage workload if such a distinction is not taken into consideration.

Since AliCloud and Systor'17_read have a certain degree of correlation, we presented the visual evidence and theoretical evidence through the autocorrelation function structure of the aggregated process of the request sequences, unfolding the existence of self-similarity. We further deployed statistical tools to estimate the Hurst parameter, and all the estimates are greater than 0.5, confirming that self-similarity occurs in AliCloud and Systor'17_read.

We designed an I/O request series generator anchored on the alpha-stable workload model, in which the parameters are directly gauged from block traces, to synthesize the temporal and spatial I/O request series for these four sets of block I/O workloads. The experimental results unveil that the proposed generator is adroit at characterizing the burstiness of the temporal and spatial I/O behaviors: our model has an accuracy edge over the conventional models.

As the storage community designs I/O scheduling, caching, or any other service policies for cloud block storage systems, such as the Alibaba Pangu [41], it is indispensable to accurately characterize I/O activities in the spatial locality. Therefore, we intend to shed the bright light on the relevance of long-range dependence in cloud block workloads from the spatial perspective in the not-too-distant future.

Acknowledgments

We want to thank the anonymous reviewers for their helpful comments in reviewing this paper. Thank Xingrui Zhang for his help in the extraction of block traces while he pursued a master's degree from the school of computer science and technology at Huazhong University of Science and Technology. Thanks also go to Jianghe Cai and Hexian Lu for their works in the major revision while they pursued their master's degrees from the department of computer science at Jinan University.

References

[1] J. Li, Q. Wang, P. Lee, and C. Shi, "An in-depth analysis of cloud block storage workloads in large-scale production," in *Proc. IEEE 16th Int. Symp. Workload Characterization*, 2020, pp. 37–47.

[2] H. Li et al., "URSA: Hybrid block storage for cloud-scale virtual disks," in *Proc. 14th Eur. Conf. Comput. Syst.*, 2019, pp. 1–17.

[3] M. Wajahat et al., "Distribution fitting and performance modeling for storage traces," in *Proc. IEEE 27th Int. Symp. Model., Anal., Simul. Comput. Telecommun. Syst.*, 2019, pp. 138–151.

[4] S. Talluri, A. Luszczak, C. Abad, and A. Iosup, "Characterization of a Big Data storage workload in the cloud," in *Proc. ACM/SPEC Int. Conf. Perform. Eng.*, 2019, pp. 33–44.

[5] S. Wang et al., "BCW: Buffer-controlled writes to HDDs for SSD-HDD hybrid storage server," in *Proc. 18th USENIX Conf. File Storage Technol.*, Santa Clara, CA, 2020, pp. 253–266.

[6] E. Xu et al., "Lessons and actions: What we learned from 10 K SSD-Related storage system failures," in *Proc. USENIX Annu. Tech. Conf.*, Renton, WA, 2019, pp. 961–975.

[7] J. Yang, S. Pei, and Q. Yang, "WARCIP: Write amplification reduction by clustering I/O pages," in *Proc. 12th ACM Int. Syst. Storage Conf.*, 2019, pp. 155–166.

[8] J. Wires et al., "Characterizing storage workloads with counter stacks," in *Proc. 11th USENIX Symp. Operating Syst. Des. Implementation*, Broomfield, CO, 2014, pp. 335–349.

[9] A. Maruf et al., "Understanding flash-based storage I/O behavior of games," in *Proc. IEEE 14th Int. Conf. Cloud Comput.*, 2021, pp. 521–526.

[10] F. Deng, Q. Cao, Y. Dong, and P. Yang, "SeRW: Adaptively separating read and write upon SSDs of hybrid storage server in clouds," in *Proc. Int. Conf. Parallel Process.*, Edmonton, AB, Canada, 2020, pp. 1–11.

[11] Z. Li et al., "Metis: Robustly optimizing tail latencies of cloud systems," in *Proc. USENIX Annu. Tech. Conf.*, Boston, 2018, pp. 981–992.

[12] C. Lorgulescu et al., "PerfIso: Performance isolation for commercial latency-sensitive services," in *Proc. USENIX Annu. Tech. Conf.*, Boston, MA, 2018, pp. 519–531.

[13] B. Seo, S. Kang, J. Choi, J. Cha, Y. Won, and S. Yoon, "IO workload characterization revisited: A data-mining approach," *IEEE Trans. Comput.*, vol. 63, no. 12, pp. 3026–3038, Dec. 2014.

[14] A. Kashyap, "Workload characterization for enterprise disk drives," *ACM Trans. Storage*, vol. 14, no. 2, 2018, Art. no. 19.

[15] R. Tinedo et al., "Dissecting UbuntuOne: Autopsy of a global-scale personal cloud back-end," in *Proc. Internet Meas. Conf.*, 2015, pp. 155–168.

[16] X. Chen, L. Rupprecht, R. Osman, P. Pietzuch, F. Franciosi, and W. Knottenbelt, "CloudScope: Diagnosing and managing performance interference in multi-tenant clouds," in *Proc. IEEE 23rd Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst.*, 2015, pp. 164–173.

[17] R. Pitchumani, S. Frank, and E. Miller, "Realistic request arrival generation in storage benchmarks," in *Proc. IEEE 31st Symp. Mass Storage Syst. Technol.*, 2015, pp. 1–10.

[18] SNIA. MSR Cambridge Traces. Received: Jan. 21, 2022. [Online]. Available: http://iotta.snia.org/traces/388

[19] C. Lee et al., "Understanding storage traffic characteristics on enterprise virtual desktop infrastructure," in *Proc. 10th ACM Int. Syst. Storage Conf.*, 2017, pp. 1–11.

[20] R. Koller and R. Rangaswami, "I/O deduplication: Utilizing content similarity to improve I/O performance," in *Proc. 8th USENIX File Storage Technol.*, 2010, pp. 211–224.

[21] D. Narayanan, A. Donnelly, and A. Rowstron, "Write off-loading: Practical power management for enterprise storage," in *Proc. 6th USENIX File Storage Technol.*, 2008, pp. 253–267.

[22] S. Gupta and A. D. Dileep, "Long range dependence in cloud servers: A statistical analysis based on google workload trace," *Computing*, vol. 102, pp. 1031–1049, 2020.

[23] Q. Li et al., "Traffic self-similarity analysis and application of industrial internet," *Wireless Netw.*, 2020, doi: 10.1007/s11276C020C02420C1.

[24] Q. Liu et al., "Self-similarity in social network dynamics," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 2, no. 1, pp. 1–26, 2016.

[25] B. Hong and T. Madhyastha, "The relevance of long-range dependence in disk traffic and implications for trace synthesis," in *Proc. IEEE Symp. Mass Storage Syst. Technol.*, 2005, pp. 1–26.

[26] C. Abad, N. Roberts, Y. Lu, and R. Compbell, "A storage-centric analysis of MapReduce workloads: File popularity, temporal locality and arrival patterns," in *Proc. IEEE 8th Int. Symp. Workload Characterization*, San Diego, CA, 2012, pp. 100–109.

[27] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar, "Synthesizing representative I/O workloads for TPC-H," in *Proc. 10th Int. Symp. High Perform. Comput. Architecture*, Madrid, Spain, 2004, pp. 142–142.

[28] R. Liu, C. Yang, and W. Wu, "Optimizing NAND flash-based SSDs via retention relaxation," in *Proc. 10th USENIX Conf. File Storage Technol.*, 2012.

[29] A. Erramilli, O. Narayan, and W. Willinger, "Fractal queueing models," in *Frontiers in Queuing*, Boca Raton, FL, USA: CRC Press, 1998, pp. 245–269.
[30] Y. Lee and J. Kim, "Characterization of large-scale SMTP traffic: The coexistence of the poisson process and self-similarity," in *Proc. IEEE 16th Annu. Meeting Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst.*, Baltimore, Maryland, 2008, pp. 143–152.
[31] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 1–15, Feb. 1994.
[32] S. Gribble, G. Manku, and E. Brewer, "Self-similarity in high-level file systems: Measurement and applications," in *Proc. SIGMETRICS Perform. Eval. Rev.*, Madison, Wisconsin, 1998, pp. 141–150.
[33] Q. Zou, D. Feng, Y. Zhu, H. Jiang, X. Ge, and Z. Zhou, "A novel and generic model for synthesizing disk I/O traffic based on the alpha-stable process," in *Proc. IEEE 16th Annu. Meeting Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst.*, Baltimore, Maryland, 2008, pp. 1–10.
[34] J. Bhimani, A. Maruf, N. Mi, R. Pandurangan, and V. Balakrishnan, "Auto-tuning parameters for emerging multi-stream flash-based storage drives through new I/O pattern generations," *IEEE Trans. Comput.*, vol. 71, no. 2, pp. 309–322, Feb. 2022.
[35] I. Norros, "On the use of fractional brownian motion in the theory of connectionless networks," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 6, pp. 953–962, 1995.
[36] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," in *Proc. Conf. Commun. Architectures, Protoc. Appl.*, 1994, pp. 269–280.
[37] Y. Kang, Y. -Y. Jo, J. Cha, W. D. Bae, W. Lee, and S. -W. Kim, "FORESEE: An effective and efficient framework for estimating the execution times of IO traces on the SSD," *IEEE Trans. Comput.*, vol. 70 no. 12, pp. 2146–2160, Dec. 2021.
[38] B. Mao, S. Wu, and H. Jiang, "Exploiting workload characteristics and service diversity to improve the availability of cloud storage systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 7, pp. 2010–2021, Jul. 2016.
[39] G. Yadgar, M. Gabel, S. Jaffer, and B. Schroeder, "SSD-based workload characteristics and their performance implications," *ACM Trans. Storage*, vol. 17 no. 1, Jan. 2021, Art. no. 8.
[40] S. Liu et al., "Analysis of and optimization for write-dominated hybrid storage nodes in cloud," in *Proc. ACM Symp. Cloud Comput.*, San Cruz, CA, 2019, pp. 403–415.
[41] S. Wang et al., "Exploration and exploitation for buffer-controlled HDD-Writes for SSD-HDD hybrid storage server," *ACM Trans. Storage*, vol. 18, no. 1, 2022, Art. no. 6.
[42] D. Zhou, W. Pan, W. Wang, and T. Xie, "I/O characteristics of smartphone applications and their implications for eMMC design," in *Proc. IEEE Int. Symp. Workload Characterization*, 2015, pp. 12–21.

Qiang Zou received the MS degree in applied mathematics from the Huazhong University of Science and Technology, in 2006, and the PhD degree in computer architecture from the Huazhong University of Science and Technology, Wuhan, China, in 2009. He is currently an associate professor with the Department of Computer Science at Southwest University. His current research interests include workload characterization, modeling, performance evaluation, and applications of deep learning.

Yifeng Zhu received the BS degree in electrical engineering from the Huazhong University of Science and Technology, China, and the MS and PhD degrees in computer science and engineering from the University of Nebraska, in 2002 and 2005, respectively. He is a professor with the Department of Electrical and Computer Engineering of the College of Engineering, University of Maine. His current research interests include computer architecture and systems, data storage systems, energy-efficient memory systems, parallel/distributed computing, embedded systems, and applications of deep learning.

Jianxi Chen (Member, IEEE) received the BE degree from Nanjing University, China, in 1999, the MS degree in computer architecture from the Huazhong University of Science and Technology(HUST), Wuhan, China, in 2002, and the PhD degree in computer architecture from HUST, in 2006. He is currently an associate professor with HUST, China. His research interests include computer architecture and massive storage systems. He has published more than 30 papers in major journals and conferences.

Yuhui Deng received the PhD degree in computer science from the Huazhong University of Science and Technology, in 2004. He is currently a professor with the Computer Science Department, Jinan University. Before joining Jinan University, he worked with the EMC Corporation as a senior research scientist from 2008 to 2009. He worked as a research officer with Cranfield University, U.K., from 2005 to 2008. His research interests cover green computing, cloud computing, information storage, computer architecture, and performance evaluation.

Xiao Qin received the BS and MS degrees in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 1992 and 1999, respectively, and the PhD degree in computer science from the University of Nebraska-Lincoln, Lincoln, Nebraska, in 2004. He is currently an alumni professor and the director of graduate programs with the Department of Computer Science and Software Engineering, Auburn University. He won the NSF CAREER Award, in 2009. His research interests include parallel and distributed systems, storage systems, fault tolerance, real-time systems, and performance evaluation.