A Novel Model for Synthesizing Parallel I/O Workloads in Scientific Applications

Dan Feng *†[1], Qiang Zou *†[2]✉, Hong Jiang ‡[3], Yifeng Zhu §[4]

* *School of Computer, Huazhong University of Science and Technology*
† *Wuhan National Laboratory for Optoelectronics, P. R. China*
[1] dfeng@hust.edu.cn, [2] hustmathcs@gmail.com

‡ *University of Nebraska-Lincoln, Lincoln, NE, USA*
[3] jiang@cse.unl.edu

§ *University of Maine, Orono, ME, USA*
[4] zhu@eece.maine.edu

***Abstract*— One of the challenging issues in performance evaluation of parallel storage systems through synthetic-trace-driven simulation is to accurately characterize the I/O demands of data-intensive scientific applications. This paper analyzes several I/O traces collected from different distributed systems and concludes that correlations in parallel I/O inter-arrival times are inconsistent, either with little correlation or with evident and abundant correlations. Thus conventional Poisson or Markov arrival processes are inappropriate to model I/O arrivals in some applications. Instead, a new and generic model based on the α-stable process is proposed and validated in this paper to accurately model parallel I/O burstiness in both workloads with little and strong correlations. This model can be used to generate reliable synthetic I/O sequences in simulation studies. Experimental results presented in this paper show that this model can capture the complex I/O behaviors of real storage systems more accurately and faithfully than conventional models, particularly for the burstiness characteristics in the parallel I/O workloads.**

I. Introduction

Understanding I/O workload characteristic is critical in system modeling and simulation-based performance evaluation. Identifying representative I/O workloads allows researchers to fairly compare existing designs and faithfully evaluate new alternative ones. Two basic approaches are widely taken to obtain representative workloads. One is to collect I/O traces in a production environment that are then carefully reconstructed during simulation [1]. The other is to use synthetic I/O requests that emulate the behaviors of actual workloads [2]. The second approach allows researchers to flexibly and efficiently study the effects of some workload parameters [3]. This paper aims to understand the parallel I/O arrival characteristics in data-intensive scientific applications and develop a generic model to accurately synthesize various parallel I/O workloads.

Cluster-based parallel I/O storage systems provide a promising approach to alleviate I/O bottleneck for scientific applications [4]. Hence, data can flow in parallel between client hosts and storage nodes without passing any centralized server. Currently, large-scale storage systems, such as Lustre [5], have been widely deployed in computing clusters for scientific applications. The ability to simultaneously execute such applications on a large number of nodes and allow a large amount of data to flow independently without passing through any centralized server results in a higher degree of data parallelism and burstiness in nodes of a large computing cluster, which motivates us to analyze and model parallel I/O workloads widely existing in scientific applications.

In this paper, we propose and evaluate a novel and generic mathematical model, the α-stable parallel I/O workload model, to accurately synthesize I/O workloads. To validate and evaluate this model, we analyze the correlations of I/O inter-arrival times, and models the parallel I/O workloads collected at the Lawrence Livermore National Laboratory (LLNL) in 2003 [6] and Los Alamos National Lab (LANL) in 2008 [7]. To the best of our knowledge, little research work conducted on this topic has been reported in the literature. We compare our model against conventional models, including the Normal, Markov, Fractional Brownian Motion (FBM) [8], and FARIMA [9] models for the parallel I/O workloads. Experimental results show that the synthetic traces generated by our model can more faithfully emulate the I/O arrival behaviors than the conventional models compared.

The rest of this paper is organized as follows. Section II gives an overview of the parallel I/O traces studied in this paper and describes the related works. Section III studies the correlations of I/O inter-arrival times and discusses the necessity of proposing a new model to accurately synthesize parallel I/O workloads with intensive burstiness. Section IV presents the α-stable I/O workload model. Section V describes the rationality of using the α-stable distribution with real traces. Section VI compares the workloads synthesized by the proposed model with real traces and others synthetic workloads. Section VII concludes this paper.

II. Background and Motivation

A. Parallel I/O Storages

A large-scale distributed parallel storage system architecture typically consists of three components: the Metadata Server (MDS), the I/O nodes and clients, as shown in Figure 1. This storage architecture is widely adopted in large scale

TABLE I

SUMMARY OF *ior2*, *f1* AND *m1* APPLICATION TRACES.

Application Traces	*ior2*			*f1*		*m1*	
	ior2-fileproc	*ior2-shared*	*ior2-stride*	*f1-restart*	*f1-write*	*m1-restart*	*m1-write*
Category	Benchmark	Benchmark	Benchmark	Physics	Physics	Physics	Physics
No. of Nodes	512	512	512	343	343	1620	1620
Trace Duration	18 sec	45 sec	202 sec	1440 sec	280 sec	249 sec	240 sec
Avg. IOs per Open	512.0	512.0	512.0	142161	1	15.3	17
Avg. IO Sizes per Open	32.8 MB	32.8 MB	32.8 MB	3993.5 MB	$\ll$ 1 MB	8.5 MB	6.5 MB

scientific applications in many institutions, such as LLNL and LANL, where some I/O traces have been collected. For LLNL's scientific applications that simultaneously run on a large number of nodes in the Lustre [5] with more than 800 dual-processor nodes, the traces collected in a Lustre cluster mainly include three parallel scientific applications: *ior2*, *f1* and *m1*, as summarized in Table I. Application *ior2* consists of three parallel I/O benchmarks, i.e., *ior2-fileproc*, *ior2-shared* and *ior2-stride*. Applications *f1* and *m1* are representative physics simulations. Both applications include two phases. While *f1* has *f1-restart* and *f1-write*, *m1* involves *m1-restart* and *m1-write*. These traces were collected in September 2003 and detailed description of these applications can be found in Ref. [10].

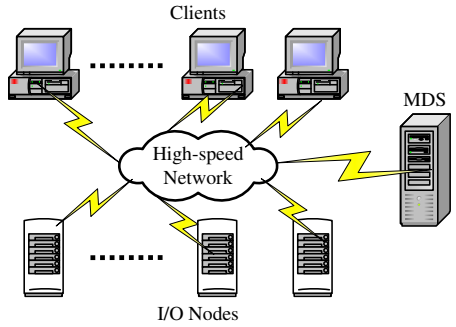

Fig. 1. Large-scale Distributed Storage System.

To test the I/O scalability for scientific applications, a benchmark called *MPI-IO* Test [7], is developed by using parallel I/O libraries at LANL. It gathers timing information for reading from and writing to file(s) using a variety of I/O profiles: (1) N processes writes to N files, i.e., *N-N*; (2) N processes writes to a shared file, i.e., *N-1-nonstrided*; (3) N processes send data to M processes that then writes to M different files or a shared file, i.e., *N-1-strided*. The traces were collected in January 2008 and the number of process used in the traces was mostly 32 or 96. Detailed description *MPI-IO* traces is given in Ref. [11].

In order to provide insights for high-performance parallel system designs, many prior research efforts have been focused on characterizing parallel I/O workload patterns [10], [12], [13]. In order to model parallel I/O workloads, prior studies usually assumed that I/O arrival process follows a Poisson distribution, and I/Os can be generated by using the Markov model [14]. However, the Markov model does not specialize in lend itself to accurately characterize the burstiness in parallel I/O workloads with very bursty I/O activities as evidenced in the LLNL and LANL application workloads, such as the *ior2* benchmark and the *f1* application described in Ref. [10]. This observation motivates us to examine the feasibility and effectiveness of using Markov model to synthesize and predict I/O requests for these scientific applications. In other words, let's consider the following key question: is it still appropriate to use a Poisson or Markov model to characterize or predict parallel I/O arrivals with the presence of intensive burstiness in scientific applications?

B. Related Work

Prior research works have focused on the studies of synthesizing I/O workload both at the disk level [3], [15], [16] and at the file system level [17]. At the disk level, the focus has been on trace synthesis [3], [15], [16] and disk access pattern identification [15], [16], [18], [19]. At the file system level, many studies provide useful insight into the design and analysis of various file systems for performance gains [10], [17]. In particular, Ref. [17] analyzes two sets of detailed, short-term application traces collected from general-purpose file systems, and finds that both exhibit self-similar like behaviors, with consistent Hurst parameters.

However, scientific applications tend to deviate significantly from commercial or generic applications in their I/O behaviors [20]. So far, several prior studies [10], [14], [21], [22] have analyzed the I/O behavior of parallel scientific applications, for tuning, managing, or optimizing parallel file systems. Ref. [14] has proposed a Markov model to synthesize and predict I/O requests for scientific applications. Ref. [10] examines the I/O burstiness of parallel I/O workloads using a simple methodology. They measure the *cumulative distribution functions* (CDF) of I/O inter-arrival times and conclude that I/O activities in the LLNL traces are very bursty in the *ior2* benchmark and the *f1* application.

While J. Oly and D. Reed have used a Markov model to predict parallel I/O requests in Ref. [14], this paper will examine the appropriateness of using the Markov model for parallel I/O in scientific applications in the following section, in light of the intensive burstiness in the four large-scale scientific application workloads collected at the LLNL and LANL.

III. CORRELATION STUDY

This section focuses on studying the correlations of I/O inter-arrival times and characterizing the I/O arrival patterns.

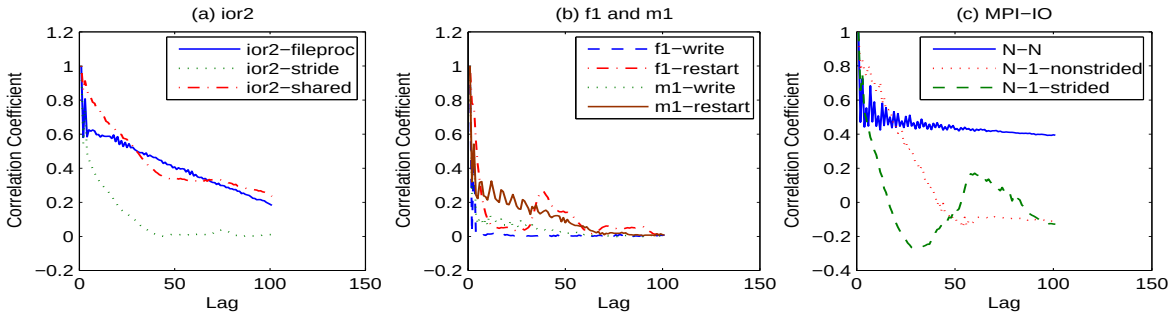

Fig. 2. From (a) to (c): auto-correlation functions (ACFs) of I/O inter-arrival times for *ior2*, *f1*, *m1* and *MPI-IO*, respectively.

This study aims to gain a deep understanding of workload behaviors, as discussed below.

In order to synthesize parallel I/O workloads and predict I/O arrivals, it is typically required to understand the workloads, particularly correlations of I/O inter-arrival times first. Auto-correlation functions (ACF) are a widely used mathematical tool to study the correlations [23], [24], i.e., by measuring if earlier values in a time sequence $X = \{x_i | i = 1, \cdots, N\}$ have some correlation to later values. The correlation coefficient at $lag\ k$ is defined as

$$c_k = \frac{1}{N-k} \sum_{i=1}^{N-k} (x_i - \bar{x})(x_{i+k} - \bar{x}) \quad (1)$$

where $\bar{x}$ is the expectation of the time series X. Then the $ACF(k)$, with a lag of k is

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

The change trends of auto-correlation coefficient can characterize the burstiness of I/O arrivals. If the correlation coefficient decreases sharply and approaches to zero quickly, then I/O requests arrive in a smooth instead of bursty fashion. In this case, there is very little or no correlation and, accordingly, independent identically distributed (IID) methods can be used to model real I/O workloads. On the other hand, if the correlation coefficient decreases very slowly, the I/O process is then very bursty and there exists a certain degree of correlations. As a result, time-series or self-similar models need to be considered to model I/O arrivals [23].

In the following, we use auto-correlation functions (ACF) to study the patterns and characteristics in I/O inter-arrival times from a time dependence perspective. The LLNL and LANL I/O traces studied in this paper are collected in many nodes. We find that the analytical results based on the traces collected on different nodes are very similar to one another in each scientific application. Therefore, this paper only presents the results of the traces collected at a randomly chosen node. The ACFs of these randomly selected traces are plotted in Fig. 2.

In Fig. 2(a), we plot the auto-correlation coefficient of I/O inter-arrival times for *ior2* as a function of lag, from $lag = 0$ to $lag = 100$. As shown in Fig. 2(a), there are strong correlations for both the auto-correlation functions of the I/O inter-arrival times in *ior2-fileproc* and *ior2-shared*. However, there is a weak correlation for I/O inter-arrival times in the *ior2-stride* workloads, but only for $lag < 40$. These observations suggest that it might be appropriate to use an independently and identically distributed (IID) method such as Markov model to synthesize the *ior2-stride* workloads, but not for the *ior2-fileproc* and *ior2-shared* workloads.

In Fig. 2(b), we plot the auto-correlation coefficients of I/O inter-arrival times as a function of lag for *f1* and *m1*, respectively. As shown in Fig. 2(b), there are evident correlations for both the auto-correlation functions of the I/O inter-arrival times in *f1-restart* and *m1-restart*. However, there is a slight correlation for I/O inter-arrival times in *f1-write* and *m1-write*, especially for *f1-write*, but only for $lag < 4$. It is reasonable to assume that the I/O arrivals in *f1-write* follow an IID process. However, the Markov model, which is independently and identically distributed, will not likely be useful in modeling the I/O requests in other workloads represented in Fig. 2(b).

In Fig. 2(c), we plot the auto-correlation coefficients of I/O inter-arrival times for *MPI-IO* as a function of lag. As shown in Fig. 2(c), there are strong correlations for the I/O inter-arrival times in the *N-N* workload. In addition, there are also evident correlations for the I/O inter-arrival times in both *N-1-nonstrided* and *N-1-strided*. Different from *N-N*, the correlations between inter-arrival times in *N-1-nonstrided* and *N-1-strided* include not only the positive correlation coefficients, but also the negative correlation coefficients. Therefore, the above observations suggest that the Markov model is not a plausible option to model I/O arrivals in *MPI-IO*.

We conclude that, in all application traces studied, there are evident correlations between inter-arrival times in subtraces collected on most computing nodes, and weak or no correlations in a small number of remaining subtraces. Examination results above show that the Markov model is inappropriate to synthesize parallel I/O workloads with intensive burstiness in scientific applications. Further, the parallel I/O workloads traced in LLNL and LANL need to be explored to provide useful insight into the synthesis of parallel I/O workloads. This motivates us to propose a more effective model to accurately synthesize the parallel I/O workloads with intensive burstiness in the next section.

IV. The α-stable Disk I/O Workload Model

I/O burstiness is very intensive in many scientific applications, such as the *ior2* benchmark and the *f1* application [10]. The salient feature of the α-stable process is its ability to represent precisely the burstiness in stochastic phenomenon, and flexibility to represent both long-range dependence and short-range dependence. This leads us to develop a new model based on the α-stable process to generate synthetic I/O requests, with a focus on faithfully emulating the burstiness that is often observed in real systems.

In most stable distributions, densities and distribution functions are not known in closed forms. Thus, α-stable distributions are generally specified by their characteristic functions.

Definition 4.1: *A random variable X is said to have a stable distribution if there exist parameters $0 < \alpha \leq 2$, $\sigma > 0$, $-1 \leq \beta \leq 1$, and $\mu \in R$ such that its characteristic function has the following form [25]:*

$$Ee^{i\theta X} = \begin{cases} e^{-\sigma^\alpha |\theta|^\alpha (1 - i\beta sign\theta \tan\frac{\pi\alpha}{2}) + i\mu\theta}, & \alpha \neq 1 \\ e^{-\sigma|\theta|(1 + i\beta sign\theta \ln|\theta|) + i\mu\theta}, & \alpha = 1 \end{cases} \quad (3)$$

where $sign\theta = \begin{cases} 1, & \theta > 0 \\ 0, & \theta = 0 \\ -1, & \theta < 0 \end{cases}$ *, α, β, σ and μ are characteristic exponent, skewness parameter, scale and location parameters, respectively.*

The characteristic exponent α represents the level of burstiness in the distribution. The distribution can be skewed if the skewness parameter β is different from zero. Variables σ and μ are called the scale and location parameters and represent the deviation and the mean of the distribution, respectively. A random variable X that follows an α-stable distribution with the above parameters is denoted by $X \sim S^\alpha_{\sigma,\beta,\mu}$ [25].

If $0 < \alpha < 2$, the characteristic function of the α-stable distribution belongs to a class of non-Gaussian functions. If $\alpha = 2$, the characteristic function will degenerate to a Gaussian function, denoted as $Eexpi\theta X = exp\{-\sigma^2\theta^2 + i\mu\theta\}$. In fact, this is the characteristic function of a Gaussian stochastic process, with a constant mean μ, variance $2\sigma^2$, and β becoming of no meaning due to $\beta \tan\pi = 0$, according to the characteristic function of the α-stable process. Therefore, by changing α, the α-stable process is able to represent the stochastic process under the Gaussian condition as well as non-Gaussian condition.

The stochastic process studied in this paper belongs to a class of the α-stable process that has both the self-similarity and the stable increments. Extending FBM under the α-stable condition, we can obtain various forms of the process, of which one is the Linear Fractional Stable Motion (LFSM) process [26]. LFSM shares all properties of the α-stable process, and its increment process is called the Linear Fractional Stable Noise (LFSN) process. The LFSN process can be expressed in a discrete domain, which makes it one of the most common mathematical modeling tools [27].

Due to the fact that realistic modeling usually exists in discrete states, the LFSN process expression in continuous states above needs to be transformed to a discrete expression, replacing the integral with a *sum* function. Through the discrete transformation of the properties of the α-stable process [25], we can express a LFSN process as a linear convolution as follows:

$$\begin{aligned} N_{\alpha,\beta,H}(i) &= (h_d * S^\alpha_{1,\beta,0})(i) \\ &= \sum_{k=1}^{Km} h_d(\frac{k}{m}) \cdot S^\alpha_{(\frac{1+\beta}{2})^{\frac{1}{\alpha}},1,0}(i - \frac{k}{m}) \\ &\quad - \sum_{k=1}^{Km} h_d(\frac{k}{m}) \cdot \tilde{S}^\alpha_{(\frac{1-\beta}{2})^{\frac{1}{\alpha}},1,0}(i - \frac{k}{m}) \quad (4) \end{aligned}$$

where $h_d(x) = \begin{cases} x^d - (x-1)^d, & x \geq 1 \\ x^d, & 0 < x \leq 1 \end{cases}$, $d = H - \frac{1}{\alpha}$, $S(i)$ is an α-stable stochastic variable that is independently and identically distributed, h_d is the discrete inner-kernel function, m is the grid parameter in the integral-discreting scheme, and K is the integral stop point. $N_{\alpha,\beta,H}(i)$ represents the discrete form of the stable LFSN process (i.e., a class of the α-stable process satisfying $\sigma = 1, \mu = 0$), $S^\alpha_{1,1,0}$ and $\tilde{S}^\alpha_{1,1,0}$ represent two independently and identically distributed discrete stochastic variables. The common distribution is $S^\alpha_{1,1,0}$. H is the *Hurst parameter* that gives a measure of the degree of self-similarity of a given time-series, $0 < H < 1$. The Hurst parameter to a set of observations can be estimated by the R/S analysis (i.e., *Pox plot*), and a detailed description of this method can be found in [28], [29].

Since the marginal distribution of a LFSN process is an α-stable process, the LFSN process has the basic properties of the α-stable process. This paper provides a novel model directly based on the LFSN process theory. According to the properties of the α-stable process, we construct an α-stable I/O workload model, and its formalization is expressed as follows:

$$IOs(i) = v \cdot N_{\alpha,\beta,H}(i) + \delta \quad (5)$$

where $IOs(i)$ represents the number of I/O requests in the i^{th} unit time, v and δ are real numbers greater than zero.

This model includes five parameters, and the physical meaning of each parameter is given as follows. α measures the degree of I/O burstiness, β represents the degree of heavy tail in the I/O traffic, H measures the degree of self-similarity, v represents the I/O mean velocity of the disk traffic, and δ represents the deviation degree relative to the I/O mean velocity of the disk traffic.

In summary, the α-stable process has a solid theoretical basis for synthesizing parallel I/O workloads. In the following section we will analyze realistic I/O traces, and then carefully scrutinize the rationality for adopting the α-stable process with credible experimental data. After that we will use real I/O traces to examine whether real trace data follow the α-stable distribution.

V. Examination of the α-stable Distribution

To examine whether I/O arrivals specified in an I/O trace follow the α-stable distribution, we first estimate the parameters of a given α-stable distribution by measuring the

TABLE II

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

Data Set	*ior2-fileproc*	α	β	σ	μ
1	p16t	0.79	1.00	0.22	1.34
2	p234t	0.88	1.00	0.29	1.31
3	p301t	0.99	1.00	0.38	1.29
4	p333t	0.85	1.00	0.26	1.33
5	p416t	0.87	1.00	0.27	1.32
Data Set	*ior2-shared*	α	β	σ	μ
6	p16t	0.98	0.42	1.33	6.76
7	p129t	0.92	0.49	1.24	6.22
8	p276t	0.96	0.52	1.25	5.69
9	p301t	0.96	0.52	1.25	5.69
10	p416t	0.97	0.38	1.24	6.79
Data Set	*ior2-stride*	α	β	σ	μ
11	p87t	0.68	0.65	0.27	2.88
12	p129t	2.00	\	0.52	3.00
13	p238t	2.00	\	0.52	3.00
14	p276t	1.55	0.94	0.48	2.88
15	p511t	0.94	0.64	0.38	2.87
Data Set	*f1-restart*	α	β	σ	μ
16	p87t	0.71	1.00	0.27	-1.16
17	p178t	2.00	\	8.78	5.00
18	p238t	0.58	1.00	0.07	-0.82
19	p278t	1.29	1.00	4.29	0.62
20	p318t	2.00	\	8.91	4.00
Data Set	*f1-write*	α	β	σ	μ
21	p3t	2.00	\	1.05	3.00
22	p15t	2.00	\	1.05	3.00
23	p33t	2.00	\	1.05	3.00
24	p36t	2.00	\	1.05	3.00
25	p46t	2.00	\	1.05	3.00
Data Set	*m1-restart*	α	β	σ	μ
26	p666t	0.66	0.65	0.53	3.78
27	p678t	1.02	1.00	1.36	2.64
28	p973t	0.81	1.00	0.83	2.76
29	p985t	1.06	1.00	1.51	2.43
30	p1028t	0.56	0.54	0.24	3.92
Data Set	*m1-write*	α	β	σ	μ
31	p345t	0.67	0.62	0.49	3.81
32	p456t	0.67	0.81	0.46	2.72
33	p567t	0.95	0.35	0.45	3.93
34	p678t	0.59	0.56	0.33	3.89
35	p789t	1.06	0.65	0.83	3.75

dataset, then compare the estimated distribution against the real distribution of the I/O traces. There are various mathematical methods to estimate the parameters of an α-stable distribution. In this paper, we choose the maximum-likelihood estimation because it is a typical method to estimate the parameters of an α-stable distribution. In addition, Quantile-Quantile (QQ) plot [30] is used to compare the estimated and real distributions.

For LLNL's scientific applications, as shown in Table I, the traces of the *ior2* benchmarks are collected on a 512-node cluster, *f1* is a large-scale physics simulation running on 343 nodes, and *m1* is an even larger physics simulation that runs on 1620 nodes. We compute the number of I/O arrivals per second. In this paper, for each of the seven workloads listed in Table I, we randomly select five data sets, for a total of 35 data sets.

First, the maximum-likelihood estimation method is used to estimate the parameter values of the α-stable distribution

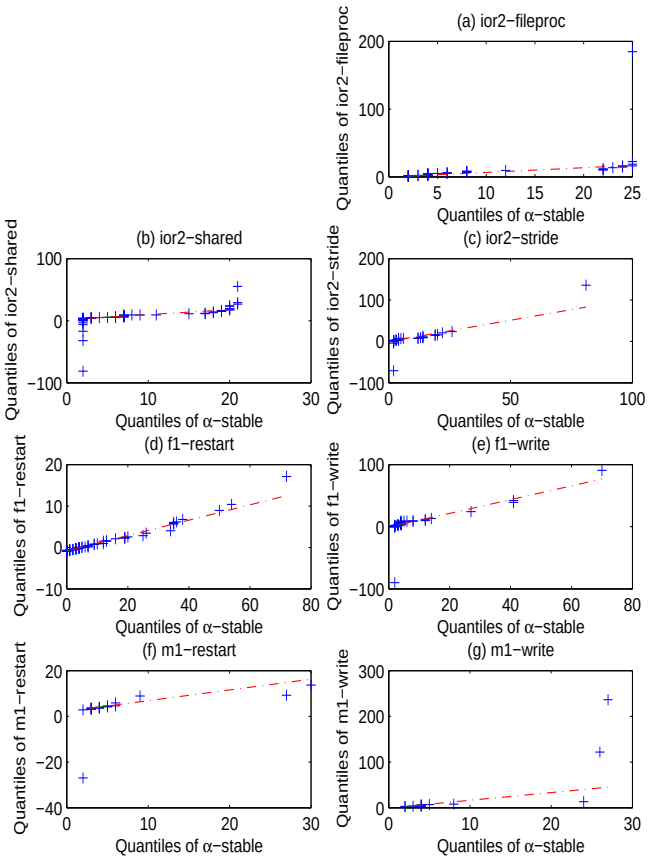

Fig. 3. From (a) to (g): QQ plots of *ior2*, *f1* and *m1* sample data versus α-stable distribution, respectively.

provided for these 35 data sets. In Table II, each row includes the sequence number of the data set, the sample data set and the estimates of four parameters. Due to the fact that the α-stable distribution will degenerate to a Gaussian stochastic process if $\alpha = 2$, with mean value μ and variance $2\sigma^2$, and β will be meaningless (see section IV). Accordingly in Table II, a slash will fill in the place of β if $\alpha = 2$. This slash symbol indicates that the relevant workload is Gaussian. According to the parameter estimates, we can obtain the relevant α-stable distribution and further check whether the given α-stable distribution is consistent with the I/O arrival process in real workloads. Due to the space constraint, this paper only discusses the results of the 3rd, 9th, 11th, 18th, 23rd, 26th, and 32nd data set in Table II.

Based on a given α-stable distribution, through the QQ plots we can judge whether the given α-stable distribution matches the probability distribution of the given data set.

In order to examine the matching degree between the α-stable distribution and the real data, the QQ plots of the given data set and the α-stable distribution are illustrated in Fig. 3. As shown in Fig. 3, the X-axis shows the quantile value of the hypothetical α-stable distribution, and the Y-axis denotes the

quantile value of the given data set. Fig. 3(a)-(g) illustrate the matching results corresponding to the seven selected data sets. Through analyzing the QQ plots, we find that a majority of data points lie along an approximate straight line. Therefore, it is reasonable to conclude that the hypothetical α-stable distribution is consistent with the real data distribution.

However, as an effective tool, the QQ plot still has some limitations in that as shown in Fig. 3(e), most of the points are compressed in a very narrow range in both dimensions, which limits our visual observation. As well, the tail of data points often fluctuate around and even beyond the theoretical straight line area, as shown in Fig. 3(b), which is induced by the accumulative effect brought by the heavy-tail distribution.

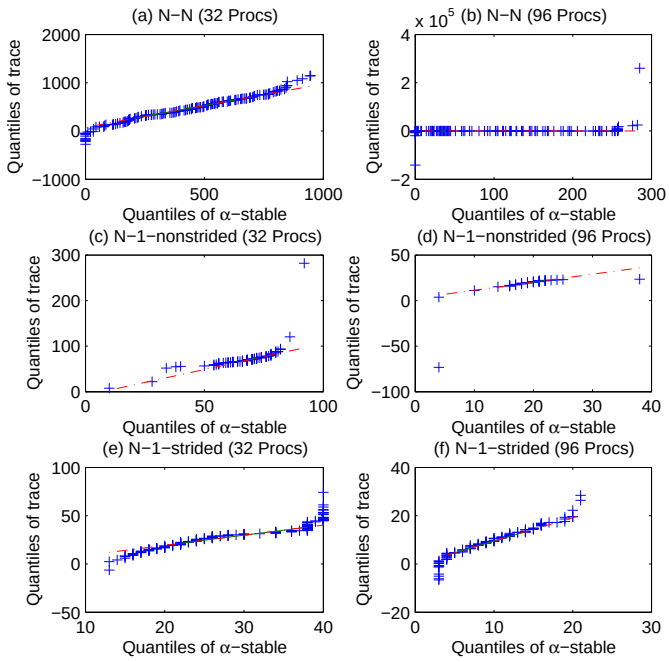

Fig. 4. From (a) to (f): QQ plots of *N-N*, *N-1-nonstrided* and *N-1-strided* sample data versus α-stable distribution, respectively.

The above methods are also used to estimate the parameter values of the α-stable distribution provided for LANL's *MPI-IO* traces and judge whether the given α-stable distribution matches the probability distribution of the given data set, respectively. Due to the space constraint, we will not summarize these parameter estimates in a table and only illustrate their QQ plots in Fig. 4, which indicates that the hypothetical α-stable distribution is consistent with the real data distribution for each sample subtrace belonging to *N-N*, *N-1-nonstrided* and *N-1-strided*, respectively.

In summary, the above analyses and comparisons clearly indicate that the hypothetical α-stable distribution is consistent with the real data distribution. Thus, we conclude that I/O arrivals in parallel I/O workloads in both LLNL's and LANL's scientific applications can be reasonably assumed to follow the α-stable distribution.

VI. Synthesizing Parallel I/O Workloads

For *ior2*, *f1*, *m1* and *MPI-IO*, we take a subset of subtraces in a random node as a testing unit. For each testing unit, we compute the I/O arrival rate, i.e., the number of I/O arrivals per second. We put all arrival rates into a group of stochastic numbers. The maximum-likelihood method is used to estimate the parameters of the α-stable process corresponding to the data sets needed to be measured. And the relevant I/O workload can be synthesized to emulate the data sets based on the constructed α-stable workload model.

The algorithm for synthesizing workload by the α-stable parallel I/O workload model is given below.

ALPHA-STABLE-SYNTHETIC-TRACE-GENERATION

INPUT: I/O mean velocity v, the deviation degree of I/O mean velocity δ, grid parameter m, integral stop point K, original trace data file *f*.
OUTPUT: an access series $(IOs(1), IOs(2), \cdots, IOs(n))$.
ALGORITHM:
for each f
  //ALPHA-STABLE-PARAMETER-ESTIMATE
  Use maximum-likelihood estimate to estimate the parameter value α, β, σ and μ of the given α-stable distribution for data sets in f;
  if $\alpha \notin (0, 2]$ *or* $\beta \notin [-1, 1]$ *or* $\sigma \leq 0$
    then break;
  else
    use Pox plot to estimate the Hurst value H
    if $H \notin (0, 1)$ or $H = 1/\alpha$
      then break;
    else calculate $d = H - 1/\alpha$, and then $h_d(x)$
      Set the initial values of m and K, and obtain $\{N_{\alpha,\beta,H}(i) : i = 1, 2, \cdots, n\}$ using Equation (4)
      Set the initial value of v and δ, and obtain $\{IOs(i) : i = 1, 2, \cdots, n\}$ using Equation (5)
end for

In order to evaluate the effectiveness of the parallel I/O workload model based on the α-stable process, this section first takes *ior2*, *f1*, and *m1* as targets to synthesize I/O workload, respectively. The synthetic workload will then be compared with real parallel I/O workloads and the workload synthesized by the conventional traffic models.

A. Analysis of Errors

According to the 35 data sets listed in Table II, we use the proposed model and conventional models to synthesize the various workloads one by one. Because a badly skewed datum in a data set can potentially render the mean of the set arbitrarily skewed from the center of the remaining data in the set, *the trimmed mean* [30] is used to evaluate the matching degrees between each real workload and the corresponding synthetic workloads. The trimmed mean of a data set is the

TABLE III

THE TRIMMED MEAN OF ERRORS FOR *ior2*.

Data Set	α-stable	Markov	Normal	FARIMA	FBM
ior2-fileproc					
1	0.56	0.88	0.98	3.41	21.09
2	0.19	0.82	0.86	2.02	13.38
3	0.72	0.62	0.91	3.12	1.01
4	0.23	0.77	0.92	3.29	9.53
5	0.53	0.91	0.67	3.10	11.75
ior2-shared					
6	0.38	0.91	0.12	6.19	0.77
7	0.05	0.88	0.09	6.89	1.03
8	0.02	0.82	0.04	6.97	18.88
9	0.15	0.02	1.18	5.49	8.87
10	0.36	0.11	0.17	6.74	7.39
ior2-stride					
11	0.16	1.11	0.25	4.89	9.21
12	0.35	0.42	0.69	3.43	0.94
13	0.19	0.59	0.58	3.91	0.70
14	0.07	0.85	0.56	3.61	1.05
15	0.04	1.25	1.35	3.80	1.50

TABLE IV

THE TRIMMED MEAN OF ERRORS FOR *f1*.

Data Set	α-stable	Markov	Normal	FARIMA	FBM
f1-restart					
16	2.02	8.15	0.85	10.63	36.63
17	1.47	6.72	0.55	11.63	18.37
18	3.05	6.31	1.67	11.25	47.23
19	4.03	2.08	0.84	10.12	5.11
20	2.13	11.01	1.94	11.45	17.7
f1-write					
21	0.01	0.13	0.03	3.65	1.88
22	0.20	0.24	0.08	3.06	2.51
23	0.02	0.13	0.05	2.65	15.94
24	0.13	0.14	0.22	2.68	5.46
25	0.02	0.07	0.03	2.63	2.37

arithmetic mean after trimming a small portion off each of the two ends of the sample data, making it more stable and resilient to abnormal data than the conventional average of samples expectation such as the arithmetic mean.

The trimmed mean of errors are illustrated in Table III, Table IV and Table V. First, we use the α-stable, Markov, Normal, FARIMA, and FBM methods to synthesize the workloads corresponding to the *ior2-fileproc* traces. As shown in Table III, in general the trimmed mean of error between the real workload and the α-stable synthetic workload is minimum, with the exception of the 3rd data set in which the trimmed mean of error for α-stable synthetic workload is slight greater than that for the Markov model. This is understandable since the α-stable model is developed to capture the essence of all workloads synthetically, not any one specific workload. Nevertheless, the matching degree of the α-stable synthetic workload for the 3rd data set is still reasonably good. In addition, for each data set, the trimmed mean of error for the FARIMA and FBM synthetic workloads are generally significantly larger than the others. This is likely due to the fact that the *ior2-fileproc* traces span only short-term time scales, while both FARIMA and FBM are self-similar models that synthesize traffics with long-range dependence. Similarly, the trimmed mean of error for *ior2-shared*, *ior2-stride* traces is also summarized in Table III.

TABLE V

THE TRIMMED MEAN OF ERRORS FOR *m1*.

Data Set	α-stable	Markov	Normal	FARIMA	FBM
m1-restart					
26	1.48	2.15	2.12	6.44	7.48
27	0.60	2.42	1.93	5.51	13.89
28	0.53	0.69	1.06	7.45	2.45
29	0.16	2.23	2.10	7.87	2.87
30	0.29	0.80	1.72	4.49	3.52
m1-write					
31	0.15	0.92	0.26	4.20	3.39
32	1.38	4.70	0.17	6.14	8.74
33	0.54	0.58	0.78	3.31	6.82
34	0.04	1.13	0.83	6.24	8.28
35	0.67	0.81	0.38	5.03	0.97

Next, we use the α-stable, Markov, Normal, FARIMA, and FBM methods to synthesize the workloads corresponding to the *f1* and *m1* traces, respectively. As can be seen from Table IV and Table V, in general the trimmed mean of error between the real workload and the α-stable synthetic workload is the minimum. And for some of the data sets in *f1* and *m1*, the trimmed mean of error between the real workload and the α-stable synthetic workload is the minimum or close to the minimum, e.g., the 16th, 17th, 18th and 20th data sets. In addition, the trimmed mean of error for the α-stable synthetic workload corresponding to the 22nd and 35th data set is close to the minimum, especially for the 22nd data set. For the 19th data set, the trimmed mean of error for the α-stable synthetic workload is larger than the workloads synthesized by the Markov and Normal methods, but only by a margin of 3.19 over the minimum error, indicating that the matching degree between the real workload and the α-stable synthetic workload is still reasonably good.

B. Empirical Study

In order to intuitively present the synthetic workloads and comparative results, without the loss of generality, we select one group of the synthetic workloads from the *ior2*, *f1*, and *m1*, respectively. The *cumulative distribution functions* (CDFs) of the selected workloads, namely, the 4th, 6th, 15th, 16th, 24th, 30th and 35th data sets, are illustrated in Fig. 5, where the X-axis shows the I/O arrival numbers per second, and the Y-axis denotes the percentage of the number of I/O arrivals. A point $(x; y)$ in the cumulative distribution curve indicates that $y\%$ of arrival rates are less than or equal to an arrival rate of x.

As can be seen from Fig. 5, the I/O workload synthesized by the α-stable model very closely matches the real trace data, especially for *ior2* and *m1*. A quantitative approach to evaluate the improvement is to analyze the error. Taking the I/O workload synthesized by the Markov model for an example, for *ior2*, *f1* and *m1*, the trimmed means of errors between the real data set and the synthesized workload through the Markov model are 0.77, 0.91, 1.25, 8.12, 0.14, 0.80 and 0.81, respectively, and the trimmed means of errors between the real data set and the synthesized workload through our

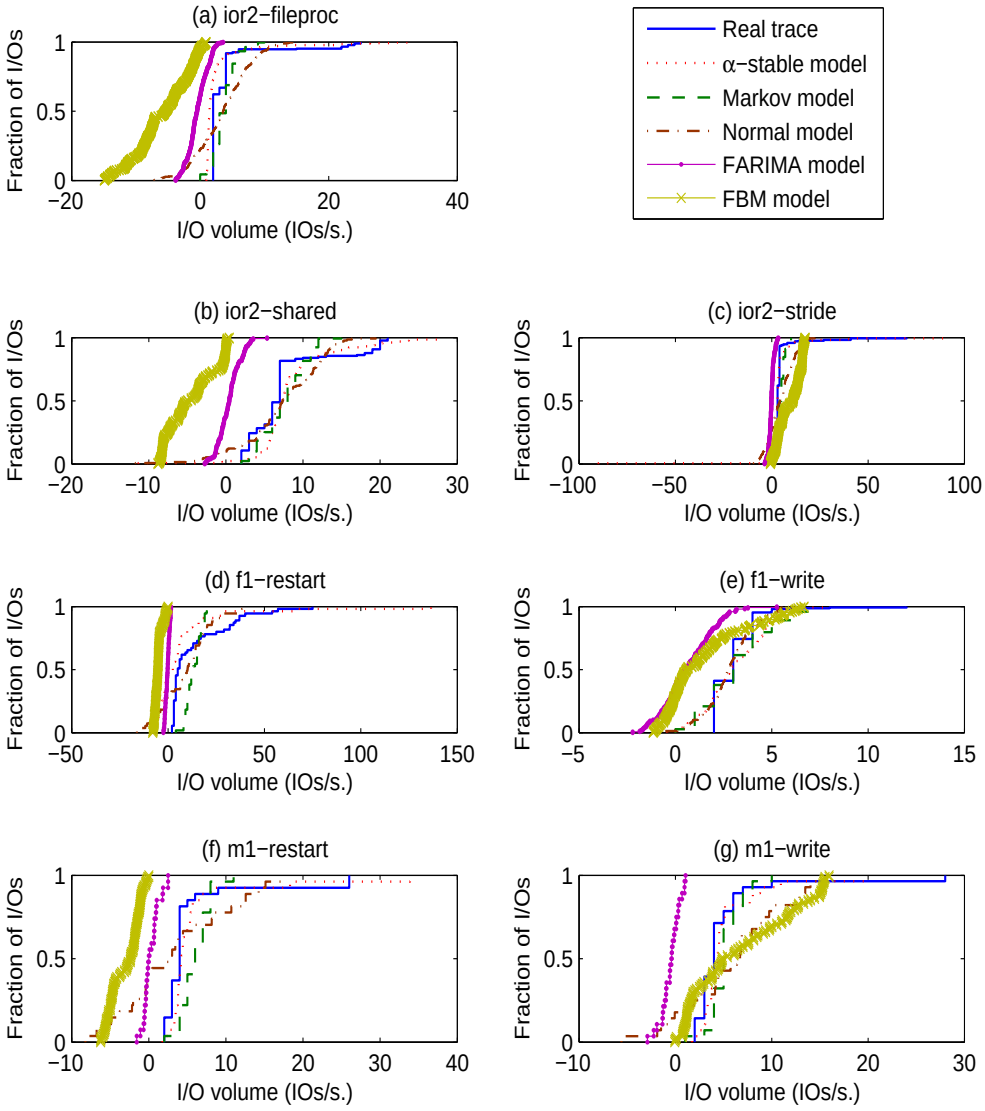

Fig. 5. From (a) to (g): CDFs of synthetic I/O traces and real traces for *ior2*, *f1* and *m1*, respectively.

proposed model are 0.23, 0.38, 0.04, 2.02, 0.13, 0.29 and 0.67, respectively. Accordingly, our proposed model can reduce the trimmed mean of error of the Markov models by 70%, 58%, 96%, 75%, 7%, 63% and 17%, respectively.

As shown in Fig. 5, the α-stable I/O workload model is by and large the best model to accurately synthesize the parallel I/O workloads in *ior2*, *f1*, and *m1*. More specifically, Table VI summarizes the relative accuracy of each synthetic model by assigning an integer value between 0 and 3 to it, where a '3' means the best matching-degree among all the synthetic workloads, a '2' means better than the average, a '1' means average, and a '0' means below the average. As can be seen from Table VI, in all cases the synthetic I/O workloads generated by the α-stable I/O workload model are more accurate than the parallel I/O workloads synthesized by the Markov model proposed in Ref. [14]. For the workloads synthesized by the Markov model, an evident deficiency is that it is difficult for the Markov model to capture the burstiness in parallel I/O workloads.

C. Synthetic Workloads for MPI-IO Benchmark

The α-stable model can also effectively synthesize I/O workload in the *MPI-IO* benchmark. We will compare all synthetic workloads generated by different models. The analysis method described previously is used here again to compare the trimmed mean of error between these synthetic workloads. Due to the space constraint, we only show the results of six subtraces, selected randomly from *N-N*, *N-1-nonstrided* and

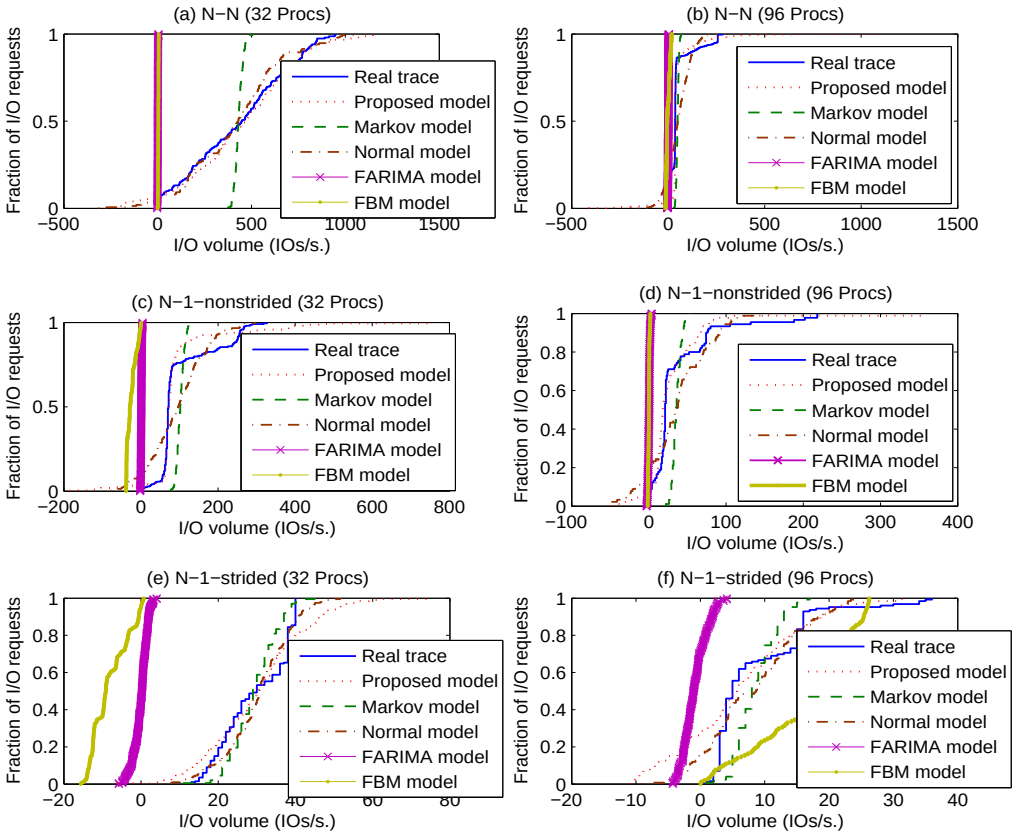

Fig. 6. From (a) to (f): CDFs of synthetic I/O traces and real traces for *N-N* , *N-1-nonstrided* and *N-1-strided*, respectively.

TABLE VI

THE MATCHING DEGREE FOR *ior2*, *f1*, *m1*.

Parallel I/O	α-stable	Markov	Normal	FARIMA	FBM
ior2-fileproc	3	2	1	0	0
ior2-shared	2	2	3	0	0
ior2-stride	3	2	2	1	1
f1-restart	2	1	3	1	0
f1-write	3	2	3	0	0
m1-restart	3	2	1	0	0
m1-write	3	2	1	0	1

TABLE VII

THE TRIMMED MEAN OF ERRORS FOR *MPI-IO*.

Data Set	α-stable	Markov	Normal	FARIMA	FBM
N-N-32	3.73	37.36	4.45	432.7	431.2
N-N-96	1.14	8.05	7.11	31.63	18.37
N-1-nonstrided-32	6.04	27.22	18.89	70.99	63.16
N-1-nonstrided-96	6.45	14.47	7.83	22.82	25.11
N-1-strided-32	0.39	0.87	0.44	31.49	25.71
N-1-strided-96	0.11	0.99	2.18	26.3	23.7

N-1-strided parallel I/O traces, respectively. Fig. 6 shows the results of our α-stable mode and other conventional models including Markov, Normal, FARIMA and FBM. The trimmed mean of errors are summarized in Table VII.

In Fig. 6, the parallel I/O workload synthesized by the α-stable model matches the real trace data very closely, especially for *N-N* (32 Procs) and *N-1-nonstrided* (96 Procs). Taking the I/O workload synthesized by the Markov model for an example, for *N-N*, *N-1-nonstrided* and *N-1-strided*, the trimmed means of errors between the real data set and the synthesized workload through the Markov model are 37.36, 8.05, 27.22, 14.47, 0.87 and 0.99, respectively. However, the trimmed means of errors between the real data set and the synthesized workload through our proposed model are only 3.73, 1.14, 6.04, 6.45, 0.39 and 0.11, respectively. Our proposed model can reduce the trimmed mean of error of the Markov models by 90%, 85%, 77%, 56%, 55% and 89%, respectively.

The proposed model can also accurately synthesize all of the parallel I/O workloads in the LANL's MPI-IO Test benchmark. And the matching degree of our proposed model is comparable to Normal model, especially in Fig. 6(a) and Fig. 6(d). However, the self-similar models such as FARIMA and FBM models can not accurately synthesize most of the real parallel I/O workloads. The Markov model proposed in Ref. [14] has an evident and critical limitation: it is difficult for the Markov model to capture the burstiness in I/O workloads.

In sum, for *MPI-IO* workloads, the α-stable model is more faithful to real-world I/O behaviors than other conventional model studied in this paper.

VII. Conclusions

The first fundamental step in finding solutions to alleviate I/O performance bottleneck in high performance computing systems is to accurately characterize the I/O demands of scientific application workload. Unfortunately, accurately modeling parallel I/O workloads remains a challenging issue due to the burstiness in the arrival process. This paper analyzes a set of real I/O traces of scientific applications running in different distributed systems. Through studying the correlations of I/O inter-arrival times in some representative parallel I/O workloads, we find that it is necessary to propose a more effective model to accurately synthesize the parallel I/O workloads with the intensive burstiness.

In this paper, we proposed and evaluated a novel and generic mathematical model, the α-stable parallel I/O workload model, to accurately synthesize I/O arrivals. We compare our model against conventional models, including the Markov, Normal, FARIMA and FBM methods. Experiment results show that the synthetic traces generated by our model can more faithfully emulate the I/O bursty arrival behaviors than the other methods. In addition, our model has five input parameters and each one has its physical meaning, allowing us to conveniently turn the I/O workload model for different environments. For example, we can change the values of parameter α so that our model can flexibly characterize burstiness under both the Gaussian and non-Gaussian conditions for parallel I/O workloads.

Acknowledgment

We would like to thank the anonymous reviewers for their helpful comments in reviewing this paper. We also thank LLNL and LANL for providing accesses to various tools and traces. Thanks go also to Xiaohu Ge, Lei Tian, Zhihua Zhou and Zhidong Wang for their help in writing this paper. This work is supported by the National Basic Research Program of China (973 Program) under Grant No.2004CB318201, the China NSF under Grant No.60503059 and No.60703046, the Program for New Century Excellent Talents in University NCET-04-0693, NCET-06-0650 and HUST-SRF No.2007Q021B, and the US NSF under Grant No.CCF-0621493, CCF-0621526, CCF-0754951, CNS-0723093, DRL-0737583.

References

[1] L. Tian, D. Feng, H. Jiang, and et al., "Pro: A popularity-based multi-threaded reconstruction optimization for raid-structured storage systems," in *Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)*, San Jose, CA, 2007.

[2] B. Anderson, "Mass storage system performance prediction using a trace-driven simulator," in *Proceedings of the 22nd IEEE Conference on Mass Storage Systems and Technologies (MSST)*, 2005.

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

[4] Y. Zhu, H. Jiang, J. Wang, and F. Xian, "A novel distributed metadata management system for large cluster-based storagev," *IEEE Transaction on Distributed and Parallel Systems*, vol. 19, no. 4, pp. 1–14, April 2008.

[5] The Lustre website. [Online]. Available: www.lustre.org

[6] Lawrence livermore national labs. [Online]. Available: www.llnl.gov

[7] Los alamos national labs. [Online]. Available: http://institutes.lanl.gov

[8] A. S. M. Priscilla and et al., "A traffic characterization procedure for multimedia applicatioins in converged networks," in *Proceedings of the 13th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'05)*.

[9] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar vbr video traffic," in *Proceedings of SIGCOMM'94*.

[10] F. Wang, Q. Xin, B. Hong, and et al., "File system workload analysis for large scale scientic computing applications," in *Proceedings of 21st IEEE Conference on Mass Storage Systems and Technologies (MSST'04)*.

[11] J. Nunez and J. Bent, "Mpi-io test users guide (version 1.0)," *Los Alamos National Labs*, May 2007.

[12] S. J. Baylor and C. E. Wu, "Parallel i/o workload characteristics using vesta," in *Proceedings of the IPPS'95 Workshop on Input/Output in Parallel and Distributed Systems (IOPADS'95)*.

[13] E. L. Miller and R. H. Katz, "Input/output behavior of supercomputing applications," in *Proceedings of the 1991 International Conference on Supercomputing*.

[14] J. Oly and D. Reed, "Markov model prediction of i/o request for scientific application," in *Proceedings of the 2002 International Conference on Supercomputing*.

[15] B. Hong and T. Madhyastha, "The relevance of long-range dependence in disk traffic and implications for trace synthesis," in *Proceedings of the IEEE Conference on Mass Storage Systems and Technologies*, 2005.

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

[17] S. Gribble, G. Manku, and E. Brewer, "Self-similarity in high-level file systems: Measurement and applications," in *Proceedings of the ACM SIGMETRICS'98*.

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

[19] M. Wang, A. Ailamaki, and C. Faloutsos, "Capturing the spatio-temporal behavior of real traffic data," in *IFIP International Symposium on Computer Performance Modeling, Measurement, and Evaluation*, 2002.

[20] S. R. Alam and J. S. Vetter, "An analysis of system balance requirements for scientific applications," in *Proceedings of the 2006 International Conference on Parallel Processing (ICPP'06)*.

[21] E. Smirni and D. A. Reed, "Workload characterization of input/output intensive parallel applications," in *Proceedings of the Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, vol. 1245, pp. 169–180.

[22] N. Tran, "Automatic arima time series modeling and forecasting for adaptive input/output prefetching," Ph.D. dissertation, University of Illinois at Urbana-Champaign.

[23] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar, "Synthesizing representative i/o workloads for tpc-h," in *Proceedings of HPCA*, 2004.

[24] A. Dainotti, A. Pescape, and G. Ventre, "Worm traffic analysis and characterization," in *Proceedings of the IEEE International Conference on Communications (ICC'07)*.

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

[26] J. Rosinski, "On the structure of stationary stable processes," *The Annals of Probability*, vol. 23, pp. 1163–1187, 1995.

[27] D. Surgailis, J. Rosinski, and et al., "Stable generalized moving averages," *Probability Theory and Related Fields*, pp. 543–558, 1993.

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

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

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