IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 22, 2019, accepted December 19, 2019, date of publication December 27, 2019,
date of current version January 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2962724

Ordinal Optimization-Based Performance Model Estimation Method for HDFS

TIAN MA[1,2], FENG TIAN[1,2], (Member, IEEE), AND BO DONG[2,3]
[1]Department of Automation Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China
[2]National Engineering Laboratory for Big Data Analytics, Xi'an Jiaotong University, Xi'an 710049, China
[3]School of Continuing Education, Xi'an Jiaotong University, Xi'an 710049, China

Corresponding author: Feng Tian (fengtian@mail.xjtu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1004500, in part by the National Natural Science Foundation of China under Grant 61877048, Grant 61721002, and Grant 61532015, in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2019JM-458, in part by the Innovation Research Team of Ministry of Education under Grant IRT17R86, and in part by the China Scholarship Council, Project of China Knowledge Center for Engineering Science and Technology, Project of Chinese Academy of Engineering The Online and Offline Mixed Educational Service System for The Belt and Road' Training in MOOC China, and Project of SERVYOU-XJTU Joint Innovation Center of Big Tax Data.

ABSTRACT Modeling and analyzing the performance of distributed file systems (DFSs) benefit the reliability and quality of data processing in data-intensive applications. Hadoop Distributed File System (HDFS) is a typical representative of DFSs. Its internal heterogeneity and complexity as well as external disturbance contribute to HDFS's built-in features of nonlinearity as well as randomness in system level, which raises a great challenge in modeling these features. Particularly, the randomness results in the uncertainty of HDFS performance model. Due to the complex mathematical structure and parameters hardly estimated of analytical models, it is highly complicated and computationally impossible to build an explicit and precise analytical model of the randomness. The measurement-based methodology is a promising way to model HDFS performance in terms of randomness since it requires no knowledge of system's internal behaviors. In this paper, the estimation of HDFS performance models on account of the randomness is transformed to an optimization problem of finding out the real best design of performance model structure with large design space. Core ideas of ordinal optimization (OO) are introduced to solve this problem with a limited computing budget. Piecewise linear (PL) model is applied to approximate the nonlinear characteristics and randomness of HDFS performance. The experimental results show that the proposed method is effective and practical to estimate the optimal design of the PL-based performance model structure for HDFS. It not only provides a globally consistent evaluation of the design space but also guarantees the goodness of the solution with high probability. Moreover, it improves the accuracy of system model-based HDFS performance models.

INDEX TERMS Distributed file system, HDFS, performance modeling, randomness, ordinal optimization.

I. INTRODUCTION

The rapid growing internet services such as Google, Yahoo!, Amazon and Facebook, are a representative style of data-intensive applications. They use big-data-oriented infrastructures like cloud computing platforms for scalable services. The Quality of Service (QoS) of the underlying distributed file systems (DFSs) contributes to the reliability and performance of these applications [1]. Thus, there is a strong demand [2]–[5] of modeling the performance of these internet-scale file systems like the Google File Systems (GFSs) [6], Amazon Simple Storage Service (S3) [7], Hadoop Distributed File System (HDFS) [8] and OpenStack Swift [9], to provide predicable performance and practical configuration guidance for performance improvement.

HDFS has emerged as a typical representative of data-intensive DFS [10]. Generally, the performance of HDFS is influenced by various factors including physical cluster deployment, disk I/O, network traffic, configuration options as well as data access patterns [11]. In a systematic view, it reveals not only nonlinear characteristics [5] but also randomness (see instances in Fig. 2 and Fig. 3, in which the observed HDFS W/R performance exhibits large range of

The associate editor coordinating the review of this manuscript and approving it for publication was Azwirman Gusrialdi.

variations with each constant input, when system configurations are the same and fixed). The randomness is mainly caused by the instability of network transmission and disk I/O [12], [13], and is increased by some performance enhancing mechanisms of HDFS, such as pipelines, caches, and load balancing [14]. In the open literature, the randomness has not been taken into account to model HDFS performance. To best of our knowledge, there is no good solution to model the randomness as well.

In the field of analyzing and modeling the performance of HDFS, there are two types of methodology. A classical methodology is to build analytical models of the system's internal architecture, components, or working mechanism (e.g., [3], [15], [16]), which significantly requires solid professional knowledge of the system. However, it is highly complicated and computationally impossible to build an explicit and precise analytical performance models in terms of the randomness of performance by using this methodology. First of all, analytical models usually have complex mathematical structures or parameters that are hardly estimated. Secondly, the professional as well as some empirical knowledge of the system is normally inadequate to figure out an appropriate analytical model due to the uncertainty caused by the randomness. Thirdly, analytical modeling typically depends on some simplifying assumptions on the system or workload behaviors, thus the accuracy of analytical models may be seriously challenged in scenarios when such assumptions are no longer satisfied [17]. The other measurement-based methodology (e.g., [4], [5], [18], [19]), which is to implement experimentation upon the system by running benchmarks, application programs, or specially designed data set, is a promising way to deal with the randomness since it requires no expertise of the dynamic behaviors of system or workloads. Evidently, the measurement-based methodology relies on the observed system behaviors which are always subject to various kinds of uncertainties due to external disturbance or modeling error. Therefore, taking the advantage of measurement-based methodology to model HDFS performance, it mainly suffers from the uncertainty of performance model with an extremely huge design space, which is caused by the randomness of performance. Heavy computational burden including time and resource costs ensues owing to the large design space.

By this means, the estimation of HDFS performance on account of the randomness is transformed to an optimization problem of finding out the real best design (a design is a solution to an optimization problem) of performance model using a limited computing budget. Since it is highly complicated to develop a precise mathematical function to describe the nonlinear characteristics and randomness of HDFS performance, piecewise linear (PL) model provides a powerful tool to approximate these features [5], [20], [21]. Then the design space is consisted of the key structural parameters of PL-based performance model. Since it is computationally impossible to find the optimal design by traversing overall design space, the core ideas of ordinal optimization (OO) are introduced to solve this problem. OO aims at using limited simulations to find some of the real good enough designs (usually defined as the real best multiple designs) from a set of observed good enough designs with high probability [22]. The major contributions of our proposed method are summarized as follows:

1. Generally, the randomness exists in every human-made complex systems like DFSs. To model the performance of DFSs, it confronts the uncertainty of performance model due to the randomness. In this paper, the estimation of performance models of DFSs on account of the randomness is transformed to an optimization problem of finding the optimal design of performance model with large design space.

2. An OO based performance model estimation method is proposed to figure out the real best design of PL-based performance model structure without occupying unreasonable computing budget. The proposed method provides a globally consistent evaluation of the design space and guarantees the goodness of solution with high probability.

3. Large experiments are conducted to verify the effectiveness of the proposed method. The accuracy of performance model built by the proposed method is improved compared with other system model-based performance models for HDFS. Moreover, the proposed method is basically a black box method that relieves developers and users from expert knowledge of system's internal behaviors, and it can be easily applied to other similar data-intensive DFSs.

In Section II, we present an overview of the OO theory. We discuss the motivation of the proposed method in Section III-A, based on which, we mathematically model the problem in Section III-B, and present specific procedure of the proposed method in Section III-C. The experimental results and analysis are illustrated in Section IV. The related work is presented in Section V. We conclude the paper in Section VI.

II. OVERVIEW OF ORDINAL OPTIMIZATION

In this section, an overview of ordinal optimization is briefly introduced in. The description of symbols in this paper are listed in Table 1.

OO was developed to deal with the optimization problems of real world complex systems [23]. These problems are often characterized by the lack of structure information, large design space, uncertainties, and they usually require time-consuming performance evaluation. As a complementary approach to optimization, OO plays a significant role to solve simulation optimization problems, and has been successfully applied in various fields, such as manufacturing systems [24], power systems [25], control systems [26], and communication systems [27].

There are two basic ideas in OO: 1) it is much easier to find out which design is better than to answer how much better in comparison; 2) nothing but the best is very costly, and small retreat in softening the goal from ''the best'' to ''good enough'' can ease the computation burden. In fact, some ''good enough'' solutions can often satisfy the

TABLE 1. Explanation of symbols.

Symbol	Explanation
Θ	Design space
N	The number of designs uniformly chosen in Θ
$\bar{\Theta}$	Sampled design space
θ_t	A design of the performance model structure
G	Good enough set
g	Size of G
S	Selected set
s	Size of S
AP	Alignment probability
k	Alignment level
T	Computing budget
$s_j(x)$	j-th submodel
swp_j	j-th SWP that divides two adjacent submodels
$p(x)$	PL-based performance model
l	The number of submodels
Ψ	Relaxed model base
$m_r(x, \varsigma)$	r-th model structure in the relaxed model base
ς	Vector of undetermined parameters of r-th model structure in the relaxed model base
M	Size of relaxed model base
ss_t	An alternative of submodel sequence
ind_j	Index of model structure in Ψ that chosen as the structure of $s_j(x)$
L	Maximal number of submodels
F	Size of sampled performance data set

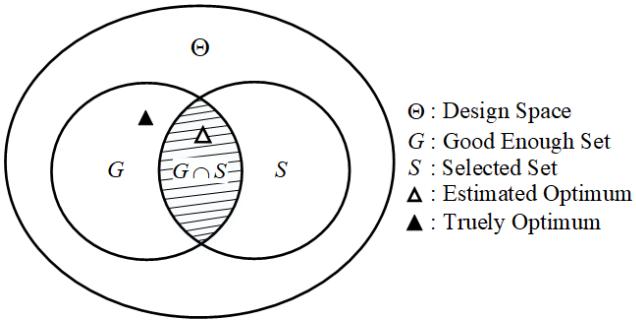
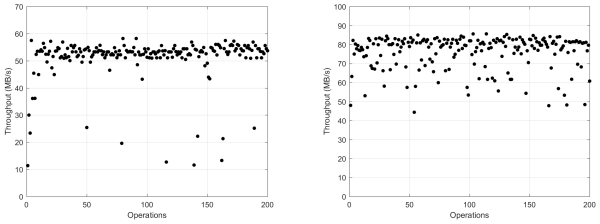


FIGURE 1. General concepts of ordinal optimization.

practical demands. In a word, OO aims at seeking the good enough with high probability instead of the best for sure. In addition, OO uses a crude but computationally fast model to estimate the performance of each design in the design space and determine the order of them. The general concepts of OO is shown in Fig. 1 [22].

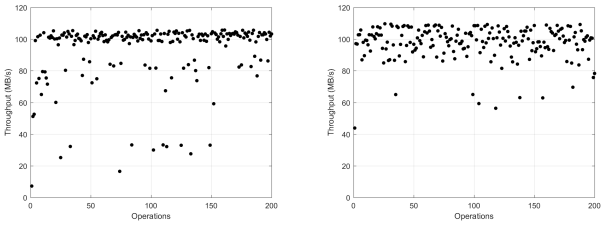
As shown in Fig. 1, Θ, the design space, is an arbitrary, huge, but finite set of alternatives that can optimize certain targeted criteria; G, the good enough set, is usually the top-g real good designs of the design space; N, the number of designs uniformly chosen in Θ; S, the selected set, is usually the estimated top-s of the N chosen designs in the design space, and it is determined by certain selection rule (SR); $G \cap S$, the truly good enough designs in S.

The size of G is denoted as $|G| = g$, and the size of S as $|S| = s$. OO ensures to build the selected set S with at least k truly good enough designs, satisfying the alignment probability, $\mathrm{AP} = \Pr(|G \cap S| \geq k)$, where k is the alignment level.



(a) Uploading a 10 MB file (b) Uploading a 100 MB file

FIGURE 2. Observed HDFS write throughput under BS64.



(a) Downloading a 10 MB file (b) Downloading a 100 MB file

FIGURE 3. Observed HDFS read throughput under BS64.

III. ORDINAL OPTIMIZATION BASED PERFORMANCE MODEL ESTIMATION METHOD FOR HDFS

A. MOTIVATION

In general, human-made systems are nonlinear with complicated structure and always subject to various kinds of uncertainties due to external disturbance as well as modeling error [21], [28]. In a systematic view, HDFS W/R performance exhibits nonlinear characteristics (like saturation and cyclical fluctuation) and randomness.

Denote the performance of HDFS as the following mapping relationship $F : x \mapsto f(x)$, where x refers to an influence factor of HDFS performance (such as the file size, block size, and the number of DataNodes). The i-th observed performance is denoted as

$$f_i(x) = \tilde{f}(x) + \xi_i \tag{1}$$

where ξ_i refers to the random noise, $\tilde{f}(x)$ represents the true performance.

In this paper, the relationship between the file size and HDFS W/R performance measured by throughput, a commonly used explicit performance QoS metric for DFS [29], under different block size settings is studied as example. In order to illustrate the randomness of HDFS W/R performance, replicated uploading and downloading experiments under the block size setting of 64 MB (denoted as BS64 for short) are conducted. The instances of sequentially uploading and downloading two files with the sizes of 10 MB and 100 MB for 200 times are shown in Fig. 2 and Fig. 3, respectively. The horizontal axis is the operation number, and the vertical axis is observed throughput, the unit of which is MB/s. Obviously, the measured HDFS W/R performance varies each time when uploading or downloading the same file with fixed system configurations.

On account of the complexity and mathematical infeasibility of constructing explicit and accurate analytical models

of the randomness, PL model is a practical and effective tool to approximately characterize the performance of HDFS taking the randomness into consideration. In brief, it is a global representation of some combination of a family of independent local submodels over the global space [20]. Each submodel within its corresponding subspace is a certain linear system model. Every two adjacent submodels are divided by a switching surface which corresponds to a switch point (SWP) of a PL-based performance model due to dimension reduction [5]. Denote a submodel as $s_j(x)$, a switch point (SWP) that divides two adjacent submodels $s_j(x)$ and $s_{j+1}(x)$ as swp_j, then a PL-based performance model is constructed as

$$\begin{aligned} &p(x) = \sum_{j=1}^{l} s_j(x), x \in \Omega \\ &s.t. \lim_{x^+ \to swp_j} s_j(x) = \lim_{x^- \to swp_j} s_{j+1}(x) \end{aligned} \tag{2}$$

where $j = 1, 2, \ldots, l-1$, $l \geq 2$; l is the number of submodels. Thus, the key structural parameters of a PL-based performance model include: 1) the number of submodel, 2) structures of submodels, and 3) values of SWPs. They turn uncertain due to the randomness.

By assuming the absence of the randomness, PL model was applied to approximately characterize the nonlinearity of HDFS W/R performance in our previous work [5]. An ordered model base was defined as an ordered set of potential model structures for submodels. It consisted of a group of linear system models with different system orders and mathematical forms caused by different distributions of zeros and poles [30]. These linear system models in the ordered model base were arranged in an ascending order based on the system order and the number of real poles.

In this case, the structure of PL-based performance model is deterministic and determined by some industrial experience and two presented strategies including the cluster quality assessment strategy and the submodel selection strategy successively. In spite of the theoretical determinacy, however, extra uncertainties due to approximation error in case of nonlinear system approximation using PL model is generated [21]. Additionally, although the strategies are effective and efficient, they are actually unable to provide guarantees of the goodness of the solutions. Concretely, first of all, the cluster quality assessment strategy to estimate the number of submodels, relies on the adopted clustering algorithms and indices to assess the quality of clustering results. Clustering is essentially a nonlinear optimization problem, which can hardly figure out the analytical solution but take the iteration solution instead [31]. The number of clusters is determined by the extremum of the optimizing index, and the reasonable division of sample data is a benchmark problem which has not found a universal evaluation index to assess the cluster quality [32]. Thus, this strategy cannot guarantee the accuracy of the estimated number of submodels. Secondly, the submodel selection strategy is essentially an empirical heuristic approach based on some industrial experience on linear and nonlinear systems to make a specific rule to determine the structure of each submodel. Hence, it is likely to choose a locally optimal submodel structure instead of the global optimum [33]. Moreover, its efficiency is facilitated by the definition of ordered model base which actually weakens the robustness of this strategy.

In this paper, by considering the impacts of randomness of HDFS performance, which result in the uncertainty of PL-based performance model structure, the key points of modeling HDFS W/R performance in system-level are subject to: 1) finding out the optimal design of PL-based performance model structure with an extremely huge design space, and 2) providing a mathematical guarantee of the goodness of the solution.

To solve this problem, a large design space of PL-based performance model is constructed on the basis of its key structural parameters. Thus, it requires great computing budget to find out the optimal design in the design space, which is very costly and impractical. Therefore, the core ideas of OO, including ordinal comparison and goal softening, are introduced to overcome this difficulty by making a compromise between the optimality of the solution and computational burden. OO contributes to not only the globally consistent evaluation of the design space but also the mathematical guarantee of the solution. The details of the proposed OO based performance model estimation method is presented in Section III-B and Section III-C.

B. PROBLEM MODELING

According to the analysis in Section III-A, modeling HDFS performance impacted by its randomness can be described as the following optimization problem:

$$\min_{\theta \in \Theta} J[p(\theta)] \tag{3}$$

where J refers to the evaluation criterion of performance model $p(\theta)$, θ represents an alternative (i.e., a design) of the structure parameter of performance model, Θ is the design space.

Denote a relaxed model base which is an unordered set as $\Psi = \{m_r(x, \varsigma)\}_{r=1}^{M}$, where $m_r(x, \varsigma)$, the r-th model structure in the relaxed model base; ς, vector of undetermined parameters; M, the size of relaxed model base. To show the correspondence between the structure of submodel and the relaxed model base, a submodel sequence (SS) is defined as

$$ss = (ind_1, ind_2, \ldots, ind_j), \tag{4}$$

where $ind_j \in \{r\}_{r=1}^{M}$, thus $s_j(x) = m_{ind_j}(x, \varsigma)$. Namely, ind_j refers to the index of model structure in Ψ that chosen as the structure of $s_j(x)$. The size of SS is $|ss| = l$.

As mentioned in Section III-A, due to the uncertainty of the model structure of PL-based performance model, l could be an arbitrary positive integer that is greater than 2. Theoretically, the model structure of $s_j(x)$ could be any linear system model. Nevertheless, it is limited to the alternatives in Ψ in this paper for simplification. Hence, the submodel sequence

is able to reflect the core parameter of PL-based performance model structure. It varies with the changes of l and ind_j. Then an alternative of SS is denoted as $ss_t = (ind_1, \ldots, ind_{l_t})$, $l_t \in \{l\}_{l=2}^{L}$, L represents the maximal number of submodels.

According to Eq. (2), to optimize the core structure parameters of PL-based performance model, the design space is constructed as

$$\Theta = \{\theta_t\}_{t=1}^{|\Theta|} \tag{5}$$

in which θ_t is defined as a design of the performance model structure

$$\theta_t = (l_t, ss_t) \tag{6}$$

where the size of the design space $|\Theta| = \sum_{l=2}^{L} M^l$.

Consequently, this optimization problem is specifically described as:

$$\min_{\theta_t \in \Theta} J = \frac{1}{F} \sum_{k=1}^{F} [f(k) - p(x(k), \theta_t, \tau_t)]^2 \tag{7}$$

where $p(x(k), \theta_t, \tau_t)$ refers to the performance model built by θ_t based on Eq. (2), $\tau_t = (swp_1, \ldots, swp_{l_t-1})$ is the set of SWPs, F is the size of sampled performance data set.

C. PROCEDURE OF THE PROPOSED METHOD

In terms of the design space constructed in Section III-B, as the increase of l and the expansion of Ψ, the size of $|\Theta|$ and the complexity of the performance model built by θ_t both exponentially increase, leading to a combinatorial problem. Therefore, it is rather difficult to figure out the optimal design via traversing the overall design space due to huge computational burden. In the proposed method, OO is applied to make a compromise between the optimality of solution and computational burden by using ''good enough'' solutions instead of the real optimal solution.

The procedure of OO based performance model estimation method for HDFS W/R performance is shown in Fig. 4. The detailed steps are described as follows.

Step 1: Set the computing budget T and determine how it is allocated to a design. Uniformly generate N sampling designs according to Eq. (6) to construct the sampled design space $\bar{\Theta}$ to scale down the full design space Θ.

It is guaranteed that $\bar{\Theta}$ is able to represent Θ [34], which means the observed top-s designs in the selected set S picked up from $\bar{\Theta}$ contains some real good enough designs in the good enough set of the full design space $G(\Theta)$ with high probability, thus it is common to evaluate $\bar{\Theta}$ instead of Θ to save the computing budget still obtaining equivalent reliable and effective solutions.

Step 2: Choose the selection rule (SR) based on T, N; set the size of the good enough set G, g; the alignment level, k; the alignment probability, AP.

Step 3: Apply the crude model to evaluate each design in $\bar{\Theta}$ to calculate the corresponding evaluation criterion by Eq. (7), and estimate undetermined parameters of the crude performance model built by each design.

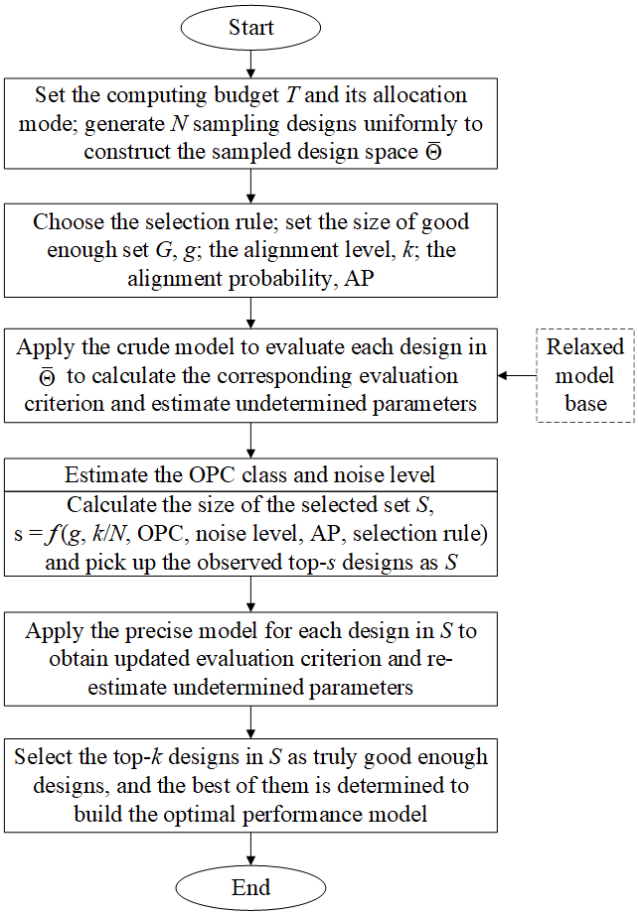


FIGURE 4. The procedure of the proposed method.

The crude model includes two parts: 1) the crude performance model that is a simplified form of Eq. (2) without the constraint; 2) the Levenberg-Marquardt (LM) algorithm [35], a widely used iterative approach to solve the nonlinear least squares problems, to estimate SWPs and undetermined parameters of the crude performance model with two user defined parameters including the maximum number of iterations, I_{max}, and the error threshold, ε. The changes of these two parameters may produce influence on the estimation results as well as the speed. In the crude model, they are set as smaller values compared with their values set in the precise model to get faster speed but crude results.

Step 4: Estimate the class of ordered performance curve (OPC) of designs in $\bar{\Theta}$ by ranking the evaluation criterion of each design, and the noise level δ, respectively. Then the size of the selected set S is determined by $s = f(g, k/N, \text{OPC}, \delta, \text{AP}, \text{SR}) = e^{\alpha} k^{\beta} g^{\gamma} + \eta$, where α, β, γ and η are constants depending on the OPC class, noise level, selection rule, and computing budget. Select the observed top-s designs as the selected set S.

Step 5: Apply the precise model for each design in S to obtain updated evaluation criterion and re-estimate undetermined parameters of the precise performance model constructed by each design in S.

Similarly, the precise model includes the precise performance model built according to Eq. (2), and the LM algorithm, in which, I_{max} is set as a larger value while ε is set as a smaller value to improve the fitting accuracy.

Step 6: Pick up the top-k designs in S as the truly good enough designs of PL-based performance model structure, and the best of them is determined to build the optimal HDFS W/R performance model.

IV. EXPERIMENTS AND ANALYSIS

The experimentation is developed on a Hadoop cluster with one NameNode and eight DataNodes in the 1 Gbps Ethernet network. The NameNode has a 2.00 GHz dual quad core Intel Xeon CPU, 8 GB RAM, and a 256 GB Seagate SAS hard drive. Each DataNode has a 2.67 GHz quad core Intel Core i5 CPU, 8 GB RAM, a 500 GB Seagate SATA hard drive to install OS and software as well as a 1 TB Seagate SATA hard drive to store HDFS data. To study the relationship between the file size and HDFS W/R performance, a file set containing 78 files with different sizes is generated. Based on prior knowledge of HDFS [2], [4], its W/R performance changes quickly and sharply for small files, and relatively turn some kind of stable as the increase of file size. Therefore, the range of size of the 78 files is from 0.25 MB to 320 MB. The sizes of small files are designed with smaller intervals like 0.25 or 0.5 MB, while large files with larger intervals like 2 or 4 MB. By uploading or downloading the file set, observed HDFS W/R throughputs over different file sizes are calculated based on the recorded uploading and downloading times of these files in the file set. The uploading and downloading experiments are deployed under four different block size settings including 32 MB, 64 MB, 96 MB, and 128 MB, respectively. These block size settings are denoted as BS32, BS64, BS96, and BS128 for short in the rest of the paper. The observed HDFS W/R throughput under the four block size settings are shown in Fig. 7(a) and Fig. 8(a), respectively.

A. INITIAL PARAMETER SETTINGS

As described in Section III-B and III-C, in order to guarantee the effectiveness of implementing the fitting algorithm, it is necessary to allocate enough data for each submodel in the experimentation. The maximal number of the submodels is set as $L = \lfloor \frac{F}{ave} \rfloor$, where $\lfloor \cdot \rfloor$ is the floor operation, F is the size of sampled performance data set, i.e., the number of files in the file set, ave is the average amount of data assigned to each submodel. Try $ave = 12$, and then $L = 6$. Therefore, the size of the full design space $|\Theta| = \sum_{l=2}^{6} 5^l = 1.9525 \times 10^4$. According to the scale of the full design space and practical experience, to greatly alleviate the computational burden, the computing budget T is set as a small amount, 500. In addition, the number of sampling designs N is also set as 500. In other words, T is allocated equally to each design once.

Since the computing budget is small, based on its allocation mode, the Horse Race (HR) selection rule is adopted to calculate s and choose the top-s selected designs as S. The reason is that HR selection rule is especially suitable and performs well in such scenario with small computing budget especially when there is no concerns on its allocation way and the width of search is more important than depth [36].

TABLE 2. Relaxed model base.

Model Index	Model Structure
1	$m_0 + m_1 e^{-px}$
2	$m_0 + m_1 e^{-px} cos(\xi x) + m_2 e^{-px} sin(\xi x)$
3	$m_0 + m_1 e^{-p_1 x} + m_2 e^{-p_2 x}$
4	$m_0 + m_1 e^{-p_1 x} + m_2 e^{-p_2 x} cos(\xi x) + m_3 e^{-p_2 x} sin(\xi x)$
5	$m_0 + m_1 e^{-p_1 x} + m_2 e^{-p_2 x} + m_3 e^{-p_3 x}$

Moreover, in crude model, the maximum iteration number of LM algorithm I_{max} is set as 200 and the error threshold ε is set as 1×10^{-8}, while they are set as 500 and 1×10^{-10} separately in the precise model.

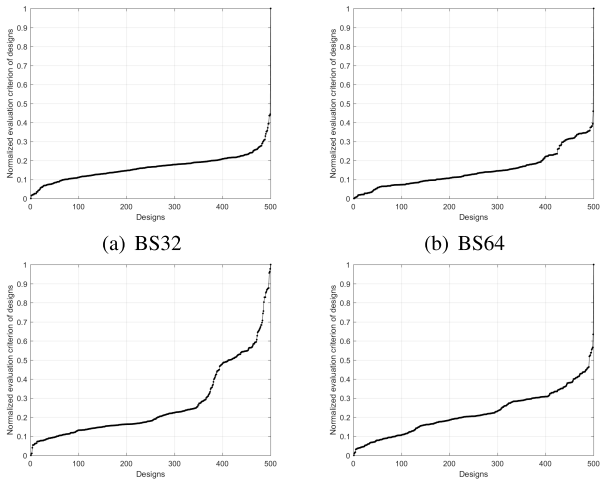
In this paper, on account of the complexity of high-order system models, the relaxed model base is consisted of five step response models of linear systems of different system orders (first-order, second-order, and third-order) with different distributions of zeros and poles, as shown in Table 2. The highest order of these system models in the relaxed model base is set as three aiming to relieve the overfitting problem as well.

B. EXPERIMENTAL RESULTS AND ANALYSIS

Based on the initial parameter settings above, the procedure of the proposed method is then implemented step by step. After Step 4, the normalized OPC curves of designs in $\bar{\Theta}$ for HDFS W/R performance models under the four block size settings are shown in Fig. 5 and Fig. 6, respectively. The horizontal axis is the ranked designs, the vertical axis is the normalized evaluation criterion (RMSE) of the designs. According to shapes of the OPC curves, the class of every OPC in Fig. 5 and Fig. 6 is Bell.

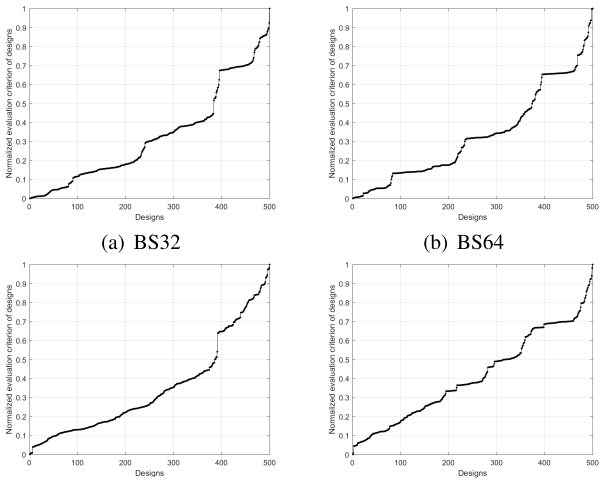
According to Eq. (1), the noise level is evaluated by calculating the average standard deviations of HDFS W/R throughput under the four block size settings. The average noise is estimated as 0.21, which is lower than the threshold of small noise level (0.25) defined in [37]. Therefore, by looking up the regressed values of the constants presented in Step 4, the size of the selected set S is calculated as $s = \lceil f(20, 1/500, \text{Bell}, \text{small}, 0.95, \text{HR}) \rceil = 20$ [37], where $\lceil \cdot \rceil$ is the ceiling operation.

Under each block size setting, the top-20 designs are then chosen as elements of the selected set S. Thereafter, according to Step 5 and Step 6, the top-1 design in S is picked up as the truly good enough design as well as the best design of PL-based HDFS W/R performance model structure. The optimal performance model is then built on the basis of Eq. (2). Consequently, the optimal designs of HDFS W/R performance model structure under the four block size settings are listed in Table 3 and Table 4, respectively. The corresponding optimal HDFS W/R performance models constructed by the optimal designs under each block size setting are shown



(a) BS32 (b) BS64 (c) BS96 (d) BS128

FIGURE 5. Normalized OPCs of designs of HDFS write performance models.



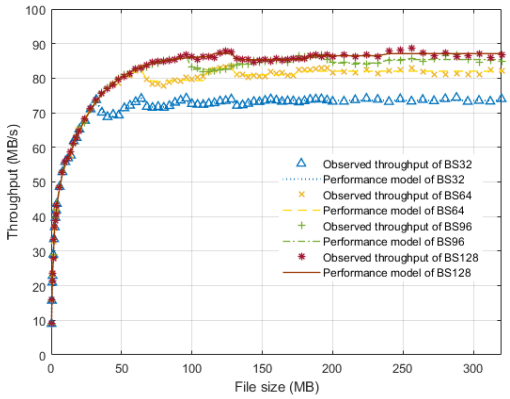
(a) BS32 (b) BS64 (c) BS96 (d) BS128

FIGURE 6. Normalized OPCs of designs of HDFS read performance models.

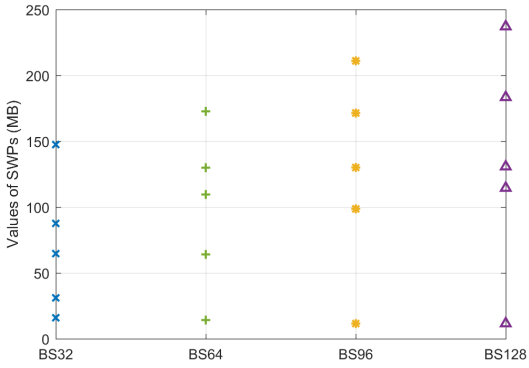
in Fig. 7(a) and Fig 8(a) separately. In which, the horizontal axis is the size of file, the vertical axis is the throughput (MB/s). The values of SWPs of each W/R performance model are shown in Fig. 7(b) and Fig 8(b). In which, the horizontal axis is the block size setting, the vertical axis is the values of SWPs (MB). For example, the five SWPs of the optimal write performance model under BS32 are marked as blue crosses, as shown in Fig 7(b).

According to Table 3, under each block size setting, the optimal HDFS write performance model consists of six submodels. And the optimal structure of all first submodels is a third-order system model with three real poles.

Similarly, as shown in Table 4, under BS32, BS96, and BS128, the optimal HDFS read performance models consist of six submodels, while the one under BS64 is consisted of five submodels. In addition, the optimal structure of every first submodel under each block size setting is a third-order system model with a real pole and two conjugate poles.



(a) Observed HDFS write throughputs and optimal performance models under each block size setting



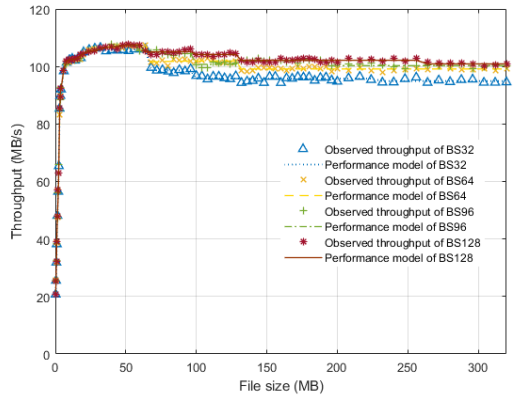
(b) SWPs of optimal write performance models under each block size setting

FIGURE 7. Observed HDFS write throughputs and performance models.

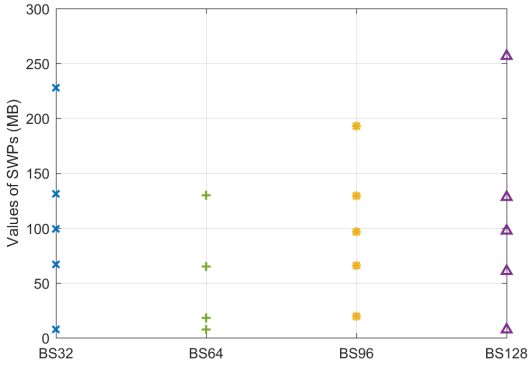
In summary, HDFS W/R performance varies observably for small files and becomes stable gradually as the increase of the file size. Particularly, the fluctuation of read performance is larger than write performance. Hence, the optimal structure of the first submodel that characterizes HDFS W/R performance for small files is estimated as a system model with the highest system order. Moreover, since the system with conjugated poles fluctuates more severely than the one with real poles, thus the optimal first submodel of HDFS write performance model has three real poles while the optimal first submodel of HDFS read performance has a real pole and two conjugate poles.

C. COMPARISON OF HDFS W/R PERFORMANCE MODELS

In this section, we mainly compare the accuracy of HDFS W/R performance model built by the proposed method with that of the one constructed by the methods in [4] and [5], respectively. For short, denote the estimated optimal performance model built by the proposed method as PerfModelByOO, the performance model established by [4] as PerfModelBySI, and the one by [5] as PerfModelByPWLM3, respectively. PerfModelByOO, PerfModelBySI and PerfModelByPWLM3 are all based on system models. The comparison of RMSEs of PerfModelByOO, PerfModelBySI, and PerfModelByPWLM3 for HDFS W/R performance are shown in Fig. 9(a) and Fig. 9(b), respectively.



(a) Observed HDFS read throughputs and performance models under each block size setting



(b) SWPs of optimal read performance models under each block size setting

FIGURE 8. Observed HDFS read throughputs and performance models.

TABLE 3. Optimal designs of HDFS write performance model under different block size settings.

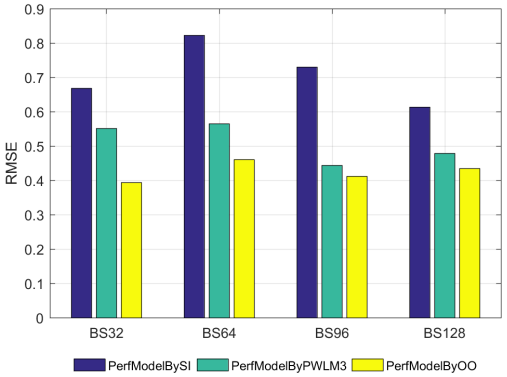
Block Size	Optimal Design	
	Number of Submodels	Submodel Sequence
BS32	6	(5,1,5,5,1,2)
BS64	6	(5,2,4,5,4,1)
BS96	6	(5,3,2,4,4,4)
BS128	6	(5,2,4,1,4,1)

TABLE 4. Optimal designs of HDFS read performance model under different block size settings.

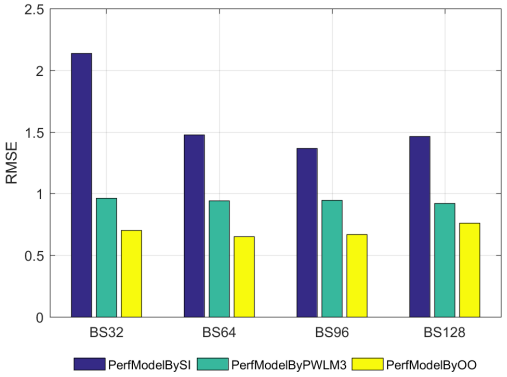
Block Size	Optimal Design	
	Number of Submodels	Submodel Sequence
BS32	6	(4,4,1,3,2,2)
BS64	5	(4,2,5,5,1)
BS96	6	(4,4,3,2,4,3)
BS128	6	(4,4,2,5,3,1)

As shown in Fig. 9(a), under each block size setting, PerfModelByOO for HDFS write performance improves the accuracy by 41.08%, 43.96%, 42.17% and 29.10% separately compared to PerfModelBySI; compared with PerfModelByPWLM3, the accuracy is improved by 28.53%, 18.38%, 7.18% and 9.20%, respectively.

Similarly, as shown in Fig. 9(b), under each block size setting, compared to PerfModelBySI, PerfModelByOO



(a) Comparison of RMSEs of PerfModelByOO, PerfModelBySI, and PerfModelByPWLM3 for HDFS write performance under each block size setting



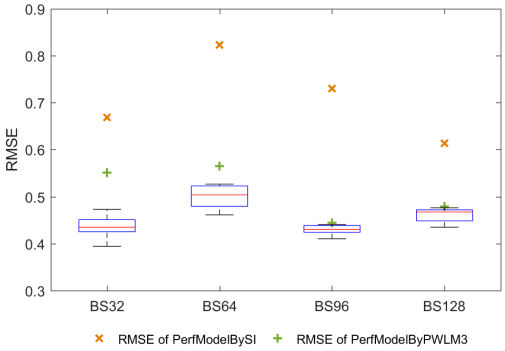
(b) Comparison of RMSEs of PerfModelByOO, PerfModelBySI, and PerfModelByPWLM3 for HDFS read performance under each block size setting

FIGURE 9. Comparison of HDFS W/R performance models: case 1.

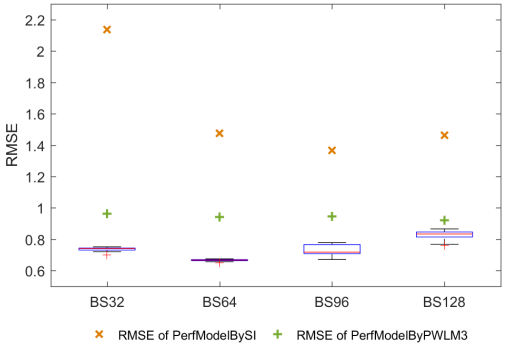
for HDFS read performance improves the accuracy by 67.18%, 55.92%, 47.86% and 47.94% separately, and by 27.21%, 30.97%, 24.67% and 17.31% in contrast to PerfModelByPWLM3.

Based on the comparison above, it is obvious that the optimal HDFS W/R performance model built by the proposed method evidently improves the accuracy of system model-based HDFS W/R performance models. Furthermore, we particularly compare the RMSEs of PerfModelByOOs for HDFS W/R performance constructed by the top-s designs (illustrated as box plot in Fig. 10) with those of PerfModelBySI and PerfModelByPWLM3 under each block size setting, shown in Fig. 10.

Moreover, as shown in Fig. 10(a) and Fig. 10(b), under each block size setting, RMSEs of PerfModelByOOs for HDFS W/R performance built by the top-20 designs in S are all smaller than those of PerfModelBySI and PerfModelByPWLM3, respectively. That is to say, the observed top-s good enough HDFS W/R performance models estimated by our proposed method are entirely better than PerfModelBySI and PerfModelByPWLM3 except for the estimated optimal HDFS W/R performance model, which provides supplementary evidence of the effectiveness of the proposed method.

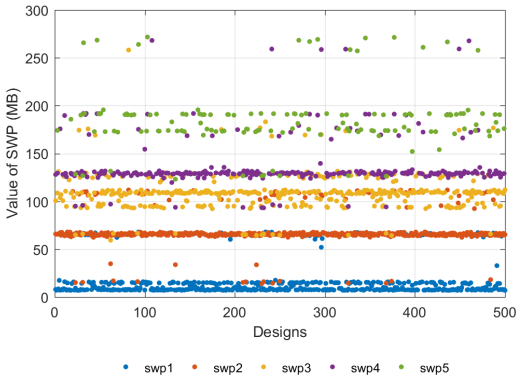


(a) Comparison of RMSEs of PerfModelByOOs built by top-s designs (drawn as box plot), PerfModelBySI, and PerfModelByPWLM3 for HDFS write performance under each block size setting

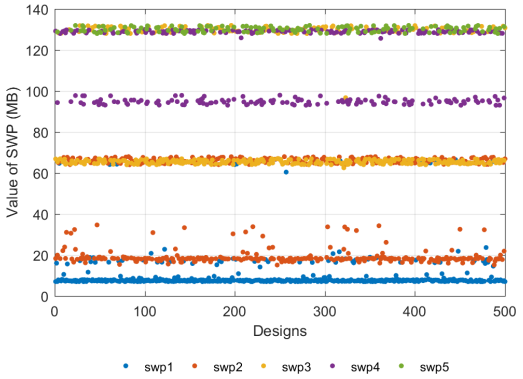


(b) Comparison of RMSEs of PerfModelByOOs built by top-s designs (drawn as box plot), PerfModelBySI, and PerfModelByPWLM3 for HDFS read performance under each block size setting

FIGURE 10. Comparison of HDFS W/R performance models: case 2.



(a) SWPs of HDFS write performance models built by all designs in $\bar{\Theta}$ under BS64



(b) SWPs of HDFS read performance models built by all designs in $\bar{\Theta}$ under BS64

FIGURE 11. Distribution of SWPs of HDFS W/R performance models under BS64.

D. ANALYSIS OF SWPS

In order to study the characteristics of SWPs, we analyze all the SWPs of HDFS W/R performance models constructed by designs in $\bar{\Theta}$ under BS64 as example. Although the number of SWPs varies for each design, we count all the SWPs in total by identifying its sequence number (for instance, j is the sequence number of swp_j) separately. The distribution of these SWPs is shown in Fig. 11. The horizontal axis is the design number, the vertical axis is the value of SWP.

As shown in Fig. 11(a), in HDFS write performance models, most of swp_1 are spread around 8, 16 or 64 MB; the majority of swp_2 are basically located in the neighborhood of 64 MB; the values of swp_3 are mainly around 96 and 112 MB; swp_4 is approximately around 128 MB, and swp_5 is distributed around either 176 or 192 MB. In addition, some of swp_4 and swp_5 are greater than 200 MB.

Similarly, as shown in Fig. 11(b), in HDFS read performance models, the majority of swp_1 are around 8 MB, while some scatter near 16 MB; most of swp_2 are located in the neighborhood of 16 and 64 MB; the values of swp_3 are mainly around 64 or 128 MB; swp_4 is distributed either near 96 or 128 MB, and swp_5 is mostly located around 128 MB.

Therefore, we may infer that the values of SWPs in a HDFS W/R performance model built by our proposed method can be characterized as $swp_j = \mu BS + 8\lambda + \zeta$, where BS refers to the block size, μ and λ are both positive integers, $\mu, \lambda \geq 0$, ζ represents the estimated deviation. This deduction is verified under BS32, BS96 and BS128 via analyzing the statistical data of all the SWPs.

V. RELATED WORK

The extant research on HDFS performance modeling and analysis typically includes two categories: The first category is the model-driven based analytical modeling methodology by abstracting a system's architecture, components, and mechanism. This methodology is also characterized as white or grey-box and usually requires solid professional knowledge of the system. Wu et al. applied the unified modeling language (UML) diagram to build analyzed performance models of DFS architecture and their key behaviors [3]. Xie et al. used communicating sequential processes (CSP) to model and analyze HDFS write and read operations [15]. Chattaraj et al. proposed a stochastic Petri net (SPN) based mathematical model to formulate the HDFS storage service activities and its dependability attributes [16]. The analytical performance models can provide explicit and illustrative description of system's internal characteristics. However, they usually have complex mathematical structures or parameters that are hardly estimated.

The other popular measurement-based methodology implements experimentation upon the system by running benchmarks, application programs, or specially designed data set. The system performance is either analyzed based on the experimental results or the comparison with other DFSs (e.g., [18], [19]). Otherwise, it is characterized by some system models driven by system identification (e.g., [4], [5]), which is basically a black-box approach and facilitates to provide support for performance QoS management via out-of-the-box control [38]. Dede et al. compared the performance of HDFS and MongoDB by running application programs [19]. Tantisiriroj et al. used micro-benchmarks and real data-intensive applications to compare the performance of HDFS and PVFS by integrating PVFS into Hadoop [18]. These approaches are able to qualitatively evaluate and analyze the performance relying on the benchmarks and applications running upon. In addition, Dong et al. implemented the system identification method to build single linear system model based performance models for HDFS in the file size domain [4]. Tian et al. proposed a piecewise linear multi-model modeling method to characterize the nonlinear characteristics of HDFS performance [5]. These approaches relieve developers and users from expertise on system's internal dynamics and are able to quantitatively characterize the relationships between system performance and parameters concerned in system level.

VI. CONCLUSION

HDFS possesses not only system-level nonlinear characteristics but also randomness attributed to the heterogeneity and complexity of its hardware and network transmission, related software operations as well as some resource conflicts in its intranet and extranet. The randomness practically results in the uncertainty of performance model for HDFS.

Since it is highly complicated and mathematically impossible to build explicit and precise analytical models of the randomness, PL model is applied to approximately characterize HDFS performance driven by the measurement-based methodology combined with system identification. The problem of estimating HDFS performance model on account of the randomness is transformed to finding out the optimal design of PL-based performance model structure with large design space using a limited computing budget. OO theory is introduced to solve this problem to find out the real best design of PL-based performance model structure for HDFS, and overcome great computational burden as well.

The experimental results show that the proposed method is effective and practical to estimate the real best structure of PL-based HDFS W/R performance model. It not only guarantees the goodness of the solution with high probability but also provides a globally consistent evaluation of the design space with small computing budget. In addition, it effectively improves the accuracy of the system model-based HDFS W/R performance model. Furthermore, the proposed method can be easily applied on other similar data-intensive DFSs since it is black box method and requires no expertise of system's internal dynamic behaviors.

In future work, more evaluation metrics in different levels will be explored to enhance the comprehensiveness of the evaluation of the design space. Furthermore, more efforts will be donated on the QoS-oriented DFS performance improvement using system theory and control theory. We will explore the usage of the performance model built by the proposed method in constructing the QoS control knobs to predict and control system performance.

REFERENCES

[1] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. Netto, and R. Buyya, "Big data computing and clouds: Trends and future directions," *J. Parallel Distrib. Comput.*, vols. 79–80, pp. 3–15, May 2015.

[2] B. Dong, Q. Zheng, F. Tian, K.-M. Chao, R. Ma, and R. Anane, "An optimized approach for storing and accessing small files on cloud storage," *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 1847–1862, Nov. 2012.

[3] Y. Wu, F. Ye, K. Chen, and W. Zheng, "Modeling of distributed file systems for practical performance analysis," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 156–166, Jan. 2014.

[4] B. Dong, Q. Zheng, F. Tian, K.-M. Chao, N. Godwin, T. Ma, and H. Xu, "Performance models and dynamic characteristics analysis for HDFS write and read operations: A systematic view," *J. Syst. Softw.*, vol. 93, pp. 132–151, Jul. 2014.

[5] F. Tian, T. Ma, B. Dong, and Q. Zheng, "PWLM^3-based automatic performance model estimation method for HDFS write and read operations," *Future Gener. Comput. Syst.*, vol. 50, pp. 127–139, Sep. 2015.

[6] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. 19th ACM Symp. Operating Syst. Princ.*, New York, NY, USA, 2003, pp. 29–43, doi: 10.1145/945445.945450.

[7] M. R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel, "Amazon S3 for science grids: A viable solution?" in *Proc. Int. Workshop Data-Aware Distrib. Comput.*, 2008, pp. 55–64.

[8] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system." in *Proc. MSST*, vol. 10, 2010, pp. 1–10.

[9] J. Arnold, *OpenStack Swift: Using, Administering, and Developing for Swift Object Storage*. Newton, MA, USA: O'Reilly Media, 2014.

[10] G. Harerimana, B. Jang, J. W. Kim, and H. K. Park, "Health big data analytics: A technology survey," *IEEE Access*, vol. 6, pp. 65661–65678, 2018.

[11] N. S. Islam, X. Lu, M. Wasi-ur-Rahman, J. Jose, and D. K. Panda, "A micro-benchmark suite for evaluating HDFS operations on modern clusters," in *Specifying Big Data Benchmarks*. Berlin, Germany: Springer, 2012, pp. 129–147.

[12] P. Čisar and S. M. Čisar, "Skewness and kurtosis in function of selection of network traffic distribution," *Acta Polytechnica Hungarica*, vol. 7, no. 2, pp. 95–106, 2010.

[13] K. Salem and H. Garcia-Molina, "Disk striping," in *Proc. IEEE 2nd Int. Conf. Data Eng.*, Feb. 1986, pp. 336–342.

[14] V. Puranik, T. Mitra, and Y. N. Srikant, "Probabilistic modeling of data cache behavior," in *Proc. 7th ACM Int. Conf. Embedded Softw.*, 2009, pp. 255–264.

[15] W. Xie, H. Zhu, X. Wu, S. Xiang, and J. Guo, "Modeling and verifying HDFS using CSP," in *Proc. IEEE 40th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 1, Jun. 2016, pp. 221–226.

[16] D. Chattaraj, M. Sarma, and D. Samanta, "Stochastic Petri net based modeling for analyzing dependability of big data storage system," in *Emerging Technologies in Data Mining and Information Security*. Singapore: Springer, 2019, pp. 473–484.

[17] D. Didona, F. Quaglia, P. Romano, and E. Torre, "Enhancing performance prediction robustness by combining analytical modeling and machine learning," in *Proc. 6th ACM/SPEC Int. Conf. Perform. Eng.*, New York, NY, USA, 2015, pp. 145–156.

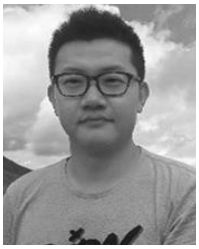
[18] W. Tantisiriroj, S. W. Son, S. Patil, S. J. Lang, G. Gibson, and R. B. Ross, "On the duality of data-intensive file system design: Reconciling HDFS and PVFS," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2011, p. 67.

[19] E. Dede, M. Govindaraju, D. Gunter, R. S. Canon, and L. Ramakrishnan, "Performance evaluation of a MongoDB and hadoop platform for scientific data analysis," in *Proc. 4th ACM Workshop Sci. Cloud Comput.*, 2013, pp. 13–20.
[20] S. A. Billings and W. S. F. Voon, "Piecewise linear identification of nonlinear systems," *Int. J. Control*, vol. 46, no. 1, pp. 215–235, Jul. 1987.
[21] G. Feng, "Controller design and analysis of uncertain piecewise-linear systems," *IEEE Trans. Circuits Syst. I. Fundam. Theory Appl.*, vol. 49, no. 2, pp. 224–232, Feb. 2002.
[22] Y.-C. Ho, "An explanation of ordinal optimization: Soft computing for hard problems," *Inf. Sci.*, vol. 113, nos. 3–4, pp. 169–192, Feb. 1999.
[23] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization of DEDS," *Discrete Event Dyn. Syst.*, vol. 2, no. 1, pp. 61–88, Jul. 1992.
[24] C. Cassandras, L. Dai, and C. Panayiotou, "Ordinal optimization for a class of deterministic and stochastic discrete resource allocation problems," *IEEE Trans. Autom. Control*, vol. 43, no. 7, pp. 881–900, Jul. 1998.
[25] S.-Y. Lin, Y. Ho, and C.-H. Lin, "An ordinal optimization theory-based algorithm for solving the optimal power flow problem with discrete control variables," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 276–286, Feb. 2004.
[26] T. E. Djaferis and D. M. Cushing, "Robust control design using stable polynomial parameterizations," in *Proc. Amer. Control Conf.*, vol. 4, 2002, pp. 2731–2736.
[27] T. Elbatt and A. Ephremides, "Optimization of connection-oriented, mobile, hybrid network systems," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 2, pp. 373–384, Feb. 1999.
[28] S. Wolfram, "Origins of randomness in physical systems," *Phys. Rev. Lett.*, vol. 55, no. 5, pp. 449–452, Jul. 1985.
[29] Q. Zhang, D. Feng, and F. Wang, "Courier: Multi-dimensional QoS guarantees for the consolidated storage system," *Future Gener. Comput. Syst.*, vol. 37, pp. 97–107, Jul. 2014.
[30] C.-T. Chen, *Linear System Theory and Design*. Oxford, U.K.: Oxford Univ. Press, 1998.
[31] T. Runkler and J. Bezdek, "Alternating cluster estimation: A new tool for clustering and function approximation," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 4, pp. 377–393, Aug. 1999.
[32] M.-S. Yang and K.-L. Wu, "Unsupervised possibilistic clustering," *Pattern Recognit.*, vol. 39, no. 1, pp. 5–21, Jan. 2006.
[33] C. Voudouris and E. P. Tsang, "Guided local search," in *Handbook of Metaheuristics*. Boston, MA, USA: Springer, 2003, pp. 185–218.
[34] S. Lin and Y. Ho, "Universal alignment probability revisited," *J. Optim. Theory Appl.*, vol. 113, no. 2, pp. 399–407, May 2002.
[35] M. I. A. Lourakis, "A brief description of the Levenberg-Marquardt algorithm implemented by Levmar," *Inst. Comput. Sci., Found. Res. Technol.*, vol. 4, no. 1, pp. 1–6, 2005.
[36] Q.-S. Jia, Y.-C. Ho, and Q.-C. Zhao, "Comparison of selection rules for ordinal optimization," *Math. Comput. Model.*, vol. 43, nos. 9–10, pp. 1150–1171, May 2006.
[37] Y.-C. Ho, Q.-C. Zhao, and Q.-S. Jia, *Ordinal Optimization: Soft Optimization for Hard Problems*. New York, NY, USA: Springer, 2008.
[38] F. Karniavoura and K. Magoutis, "Decision-making approaches for performance QoS in distributed storage systems: A survey," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 8, pp. 1906–1919, Aug. 2019.

TIAN MA received the B.S. degree in automation science and technology from Xi'an Jiaotong University, in 2012, where she is currently pursuing the Ph.D. degree with the National Engineering Laboratory of Big Data Analytics. She was a Visiting Student with the University of Chicago, from 2017 to 2018. Her research interests focus on cloud storage and complex systems modeling.

FENG TIAN (Member, IEEE) received the B.S. degree in industrial automation and the M.S. degree in computer science and technology from the Xi'an University of Architecture and Technology, Xi'an, China, in 1995 and 2000, respectively, and the Ph.D. degree in control theory and application from Xi'an Jiaotong University, Xi'an, in 2003.

He has been with Xi'an Jiaotong University, since 2004, where he is currently with the National Engineering Laboratory of Big Data Analytics and with the Systems Engineering Institute, as a Professor. He is a member of the Satellite-Terrestrial Network Technology Research and Development Key Laboratory, Shaanxi. His research interests include cloud computing, big data analytics, learning analytics, and system modeling and analysis. He was a recipient of the Second Class Prize of the National Science and Technology Progress Award of China (fourth position, in 2017), the First Class Price of the MOE Science and Technology Progress Award of China (fourth position, in 2015), the First Class Prize of the Science and Technology Progress Award of Chinese Electronical Society (fifth position, in 2013), and the TOP Grade Prize of Teaching Achievement Award of Shaanxi Province Award, China (second Position, in 2017). He is a member of CCF.



BO DONG received the Ph.D. degree in computer science and technology from Xi'an Jiaotong University, in 2014. He did his Postdoctoral Research with the MOE Key Laboratory for Intelligent Networks and Network Security, Xi'an Jiaotong University, from 2014 to 2017, where he currently serves as the Research Director of the School of Continuing Education. His research interests focus on cloud computing, intelligent e-Learning, and data mining.