Tier-Scrubbing: An Adaptive and Tiered Disk Scrubbing Scheme with Improved MTTD and Reduced Cost

***Abstract*—Sector errors are a common type of error in modern disks. A sector error that occurs during I/O operations might cause inaccessibility of an application. Even worse, it could result in permanent data loss if the data is being reconstructed, and thereby severely affects the reliability of a storage system. Many disk scrubbing schemes have been proposed to solve this problem. However, existing approaches have several limitations. First, schemes use machine learning (ML) to predict latent sector errors (LSEs), but only leverage a single snapshot of training data to make a prediction, and thereby ignore sequential dependencies between different statuses of a hard disk over time. Second, they accelerate the scrubbing at a fixed rate based on the results of a binary classification model, which may result in unnecessary increases in scrubbing cost. Third, they naively accelerate the scrubbing of the full disk which has LSEs based on the predictive results, but neglect partial high-risk areas (the areas that have a higher probability of encountering LSEs). Lastly, they do not employ strategies to scrub these high-risk areas in advance based on I/O accesses patterns, in order to further increase the efficiency of scrubbing.**

We address these challenges by designing a Tier-Scrubbing (TS) scheme that combines a Long Short-Term Memory (LSTM) based Adaptive Scrubbing Rate Controller (ASRC), a module focusing on sector error locality to locate high-risk areas in a disk, and a piggyback scrubbing strategy to improve the reliability of a storage system. Our evaluation results on realistic datasets and workloads from two real world data centers demonstrate that TS can simultaneously decrease the Mean-Time-To-Detection (MTTD) by about 80% and the scrubbing cost by 20%, compared to a state-of-the-art scrubbing scheme.

I. Introduction

Hard disks are widely used as primary storage devices in modern data centers. A disk failure can fall into two categories: device level (i.e., a complete disk failure) and block level (i.e., a partial disk failure) [1]. A complete disk failure can lead to temporary data loss and thus system unavailability. It can furthermore result in permanent data loss if the lost data cannot be recovered by provisioned data protection schemes (e.g., replication and erasure codes [2], [3]). The failures of disks at a block level (referred to as latent sector errors (LSEs)) also impacts data reliability and availability. According to our statistics on disks from a real world data center, about 35% of disk failures are caused by LSEs.

Compared with traditional passive fault tolerance techniques like EC (Erasure Codes, a solution for complete disk failures), RAID (Redundant Arrays of Independent Disks, a solution for complete disk failures) [4], and intra-disk redundancy (a solution for LSEs), proactive failure prediction aims to ensure the reliability and availability of large scale storage systems by taking preventive measures before failures actually happen.

Nowadays, researchers have proposed machine learning (ML)-based methods to predict complete disk failures [5]–[9] based on self-monitoring, analysis and reporting technology (S.M.A.R.T) data [10], which achieved good predictive performance. However, these approaches still suffer from a 1% false positive rate (FPR, the proportion of good disks that are falsely predicted as failed ones), which makes it difficult to put them into production in real data centers. Even though 1% sounds small, it will result in high costs in a modern large scale data centers with hundreds of millions of disks, because all the wrongly predicted disks will have to be replaced. As LSE problems become more prominent, many researchers have focused on the prediction of LSEs using ML, based on S.M.A.R.T data [1], [11]. Although the predictive results (about 10% FPR) are not on par with the results for complete disk failures, these methods do not force the data center operator to replace the complete disk in case of (potentially incorrectly) predicted LSEs; instead only apply simple verification operations (i.e., disk scrubbing, etc.) have to be applied. The reason is that a sector error is much easier to verify than a complete disk failure. Mahdisoltani et al. [11] used a random forest-based approach [12] to predict LSEs, with a scheme that accelerates the scrubbing with a fixed rate when LSEs are predicted. Jiang et al. [1] proposed the Scrub Unleveling (SU) scheme, (again based on random forests) to improve the cost-efficiency of disk scrubbing. Instead of trading off scrubbing cost and data reliability, they aim to achieve a low scrubbing cost and high data reliability at the same time in a storage system.

However, these studies have several limitations. From the perspective of predictive models, **(1)** these methods take a single snapshot of S.M.A.R.T attributes as training data for prediction, without considering the sequential dependency between different statuses of a hard disk over time (which was observed by many researches [9], [13]); **(2)** these models only apply binary classification to the status of a sector, i.e., they predict whether there will be LSEs within the next few days or not, and cannot quantify the temporal distance to the actual sector error occurrence. Therefore, they only accelerate the scrubbing at a fixed rate (i.e., $\times 2$), which can result in unnecessary scrubbing cost (as we discuss in detail in Section II-B1). From the perspective of a scrubbing strategy, **(3)** the existing approaches simply uniformly increase the scrubbing rate for the full disk without giving special consideration to high-risk areas (the areas that have a higher probability of encountering LSEs in a disk), which could be more important to improve the reliability of the storage system; **(4)** they also do not explore how to scrub these high-risk areas in advance based on I/O accesses, in order to further accelerate the efficiency of disk scrubbing.

In this paper, we propose a novel scrubbing scheme called Tier-Scrubbing (TS). TS is an adaptive and effective scrubbing scheme that combines an ML-based adaptive scrubbing rate controller, a module focusing on sector error locality to locate high-risk areas in a disk, and a piggyback scrubbing strategy based on I/O accesses. Our goal is to achieve a lower MTTD accompanied by a decrease in the scrubbing cost, in order to increase the reliability of a large scale storage system (compared to a state-of-the-art scrubbing scheme).

We make the following contributions:

- **Solve the limitation (1) and (2):** at the disk level, we propose an Adaptive Scrubbing Rate Controller (ASRC) based on the LSTM algorithm [14] to not only predict LSE disks and non-LSE disks, but also to quantify the sector error risk. We model sequential dependencies within the training data, and accelerate the scrubbing of these LSE disks at an adaptive (non-fixed) rate based on the predictive results (Section II-B1).
- **Solve the limitation (3):** we explore the locality of sector errors by analyzing a large number of disks. Based on our findings,

we further study the partial high-risk areas of the LSE disks and focus on them so as to increase the reliability of the storage system at the sector level (Section II-B2).

- **Solve the limitation (4):** for identified high-risk areas, we propose a piggyback scrubbing strategy (at the level of I/O operations). We opportunistically optimize the scrubbing strategy to further improve the efficiency of scrubbing based on I/O accesses (Section II-B3).
- We experimentally evaluate our proposed scrubbing scheme on datasets and workloads from two real world data centers. We find that our scheme can achieve a higher reliability with a lower scrubbing cost, compared to a state-of-the-art scrubbing scheme, in all cases (Section III-B2).

II. Overview of Tier-Scrubbing

In this section, we first provide an overview of our proposed disk scrubbing scheme TS in Section II-A, and then introduce the approach on different levels in Section II-B.

A. Scheme Overview

Figure 1 provides an overview of our proposed scrubbing scheme TS which combines an LSTM-based adaptive scrubbing rate controller (ASRC), a module focusing on sector error locality to locate high-risk areas in a disk, and a piggyback scrubbing strategy based on I/O accesses. Disk scrubbing aims to find LSEs as early as possible, and to perform sector repair or reallocation in advance before a sector error is encountered by an application. In large data centers, different disks have different probabilities of sector errors, which disqualifies a fixed scrubbing rate for all disks. We propose an Adaptive Scrubbing Rate Controller (ASRC) that contains an LSTM-based model (LSTMs [15] are a special kind of recurrent neural networks, capable of learning long-term dependencies) to predict the sector risk degree (e.g., in range of 0-7) for all disks in the data center, based on a series of observed S.M.A.R.T data. We subsequently leverage our risk prediction to determine an adaptive disk scrubbing rate at the **disk level**.

We refer to disks predicted as LSE disks (having a risk greater than 0) with higher (lower) risk degrees as high-risk (low-risk) disks (the lowest degree is 1 and the highest is 7). These disks employ different scrubbing rates based on different sector risk degrees. Disks predicted as non-LSE disks (risk degree 0) perform the scrubbing analogous to the scheme SU (i.e., decreasing the fixed scrubbing rate). To the best of our knowledge, we are the first to propose an LSTM-based approach to predict the disk sector risk degree rather than just modeling this task as a binary classification problem.

In general, sequential disk scrubbing operates in a sequential manner to scrub sectors without specific priorities. Our experimental results show that the probability of the occurrence of sector errors in a single disk is not evenly distributed, but has peaks around certain localities. The probability that a new sector error occurs around a previously observed sector errors of a disk is much higher than in other areas. Therefore, we focus on these high-risk areas of LSE disks so as to scrub these areas with a higher priority at the **sector level**. Note that we scrub the disk with an adaptive accelerated scrubbing rate for a full disk, if the predicted LSE disk has no previously observed sector errors.

We furthermore propose a piggyback scrubbing strategy to scrub high-risk areas at the **level of I/O operations**. When the application I/O operations access these areas, we execute scrubbing (i.e., a piggyback read operation, as detailed in Section II-B3) in the untouched fragmented sectors. Note that we perform sequential scrubbing with

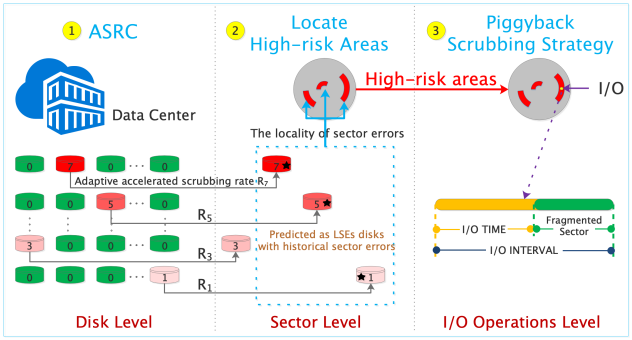


Fig. 1. The overall scheme of TS which combines an LSTM-based adaptive scrubbing rate controller (ASRC), a module focusing on sector error locality to locate high-risk areas in a disk, and a piggyback scrubbing strategy based on I/O accesses.

an adaptive accelerated rate (the scheme SU applies the sequential scrubbing with a fixed accelerated rate) on the LSE disk, and that the piggyback scrubbing strategy is only executed if the application I/O operations access the high-risk areas of the LSE disk.

B. Approach

TS is a scrubbing mechanism to address the challenges described in Section I. It predicts the sector risk degree of an LSE disk, locates the high-risk areas, and opportunistically optimizes the scrubbing strategy with the goal to decrease the MTTD and scrubbing cost. In the following, we describe TS in detail on three different levels: disk, sector and I/O operations.

1) **ASRC at Disk Level***:* The goal of ASRC is to build a model to predict the sector risk degree of a disk and accelerate the scrubbing of LSE disks with an adaptive rate. Figure 2 illustrates the architecture of ASRC, which comprises of three parts: the LSTM-based predictor, the scrubbing rate controller and a sample pool.

LSTM-based predictor. Almost all hard disk drives provide S.M.A.R.T data, which directly or indirectly reflects the health of disks and even contains some statistical information. Each S.M.A.R.T attribute entry is a quintuplet (ID, Normalized, Raw, Threshold, Worst). Not all elements in this tuple are commonly used. Most research leverages the first three elements (ID, Normalized, and Raw) [1], [5], [11]. For a fair comparison, we also use these three elements from our collected datasets. In addition, many researchers [1], [16]–[18] apply classical ML methods like SVM, Decision Tree, Random Forest or Logistic Regression based on the S.M.A.R.T attributes as training samples. However, some research has shown that the S.M.A.R.T signal of the disk changes dynamically with a certain trend [9], [13], thus the current state of the disk may depend on a long-term historical trend. In contrast to all the aforementioned works, we apply an LSTM-based predictor to incorporate sequential information when predicting the sector risk degree of a disk. LSTMs have been successfully applied to a variety of applications, including text sentiment classification [19] and multi-language text classification [20].

We designed an N-to-1 LSTM model showed in the first part of Figure 2 (where N denotes the number of input neurons) which consists of an input layer I, three hidden layers L, two dense layers M and an output layer D. The data which we feed to the network is comprised of a vector $I(t)$ that represents the S.M.A.R.T attributes at time t. In contrast to traditional neural networks, the LSTM operates over sequences of input vectors. This structure is able to capture the historical context of health statuses and makes LSTMs suitable for

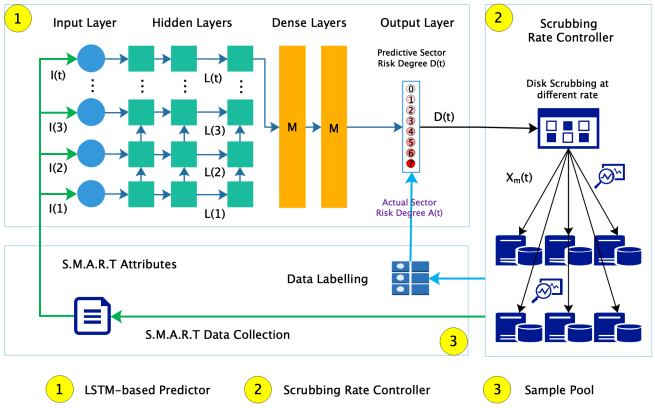


Fig. 2. The architecture of ASRC which contains an LSTM-based predictor, a scrubbing rate controller and a sample pool.

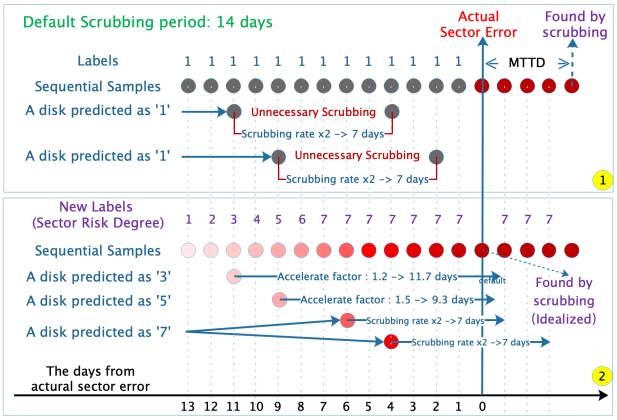


Fig. 3. The different data labelling methods. The part one on the top of this figure describes the data labeling method in existing work [1], [11], while the part two on the bottom of this figure describes our new data labeling method.

tasks related to sequential prediction. The output of the hidden layer at time t is $L(t)$, and it maintains an internal representation of the history of S.M.A.R.T attributes. The output of the network at time t is $D(t)$, which is a vector (the length of the vector is equal to the range of possible degrees) that represents the sector risk degrees (the lowest degree is 0 and the highest is 7) of the disk. We employ a softmax output to guarantee that $D(t)$ is a proper probability distribution over the risk degrees. We choose the corresponding risk degree with the highest probability in the output. Note that we consider a disk predicted to be 0 as a non-LSE disk. Our new predictor is able to both predict whether a disk will have a sector error or not (a binary classification) and the sector risk degree (multiclass classification) using sequential information. We apply the $tanh$ activation function to the outputs of hidden layers, and train our network to minimize the cross entropy [21] between observed and predicted samples (the standard loss for multiclass classification problems).

Sample Pool. Our sample pool consists of collected S.M.A.R.T data and the corresponding labels showed in the third part of Figure 2. We collect the S.M.A.R.T data of each disk in the data center every day, but the scrubbing is issued once every scrubbing period (analogous to the scheme SU). The S.M.A.R.T data we collected is recorded as a vector, $I(t) = \{a_{t1}, a_{t2}, a_{t3}, ..., a_{tn}\}$, where n is the number of attributes (we refer to Section III-A1 for the detailed attributes that we use). In order to have a fair comparison with the scheme SU and the pragmatic baseline of scanning the entire disk drive once every two weeks [11], [22], we also set the scrubbing period to 14 days. Before we present our own data labeling method, we quickly review the methods applied in existing work [1], [11] and illustrate them in part one on the top of Figure 3. For a disk with an observed sector error, they label all samples in a period of two weeks (one scrubbing period) before the occurrence of the actual sector error with 1, and label the non-LSE samples with 0. The disadvantage of this labeling method is that all the samples labeled as 1 are treated the same even though they have different time distances to the actual sector error occurrence. For example, a disk might predicted as 1 even though it is 11 days away from having an actual sector error. As a result, the disk will be exposed to accelerated scrubbing by a given fixed factor (i.e, $\times 2$) and this disk scrubbing might be completed within 14/2 = 7 days. However, in this period, no sector error actually occurs, so this approach would have introduced unnecessary additional scrubbing cost. In order to decrease the unnecessary scrubbing cost, we propose a new data labeling method, as shown in part two on the bottom of Figure 3. We introduce different sector risk degrees, ranging from 1 to 7 for the samples, based on their time distance to the actual sector error occurrence. The higher the degree, the higher the risk of sector errors (e.g., the closer we are to the error occurrence). Note that all samples are labeled with 0 for error-free disks.

Scrubbing Rate Controller. The purpose of our scrubbing rate controller is to set an adaptive scrubbing rate for a disk based on the predicted sector risk degree showed in the second part of Figure 2. The ideal situation would be that a sector error is immediately found after occurrence by the scrubbing. We therefore set an adaptive scrubbing rate for a disk predicted as having LSEs to ensure that the actual sector error has a chance to be found during the subsequent scrubbing period. Since we set the maximum accelerated scrubbing rate to a factor of 2 (resulting in an accelerated scrubbing period of 7 days) in our real world system, we label all samples from 7 to 0 days before the actual sector error occurrence with the same highest degree 7 showed in the second part of in Figure 3. For the labels 1 to 6, the adaptive scrubbing acceleration factor is calculated as follows:

$$X_m(t) = \frac{\lfloor \frac{P}{P - D_m(t)} \cdot 10 \rfloor}{10} \quad (1)$$

where m denotes the number of disks, P is the default scrubbing period (i.e., 14 days), and $D_m(t)$ refers to the predicted sector risk degree of disk m at time t. Note that the purpose of the $Floor$ operation $\lfloor \rfloor$ in Equation 1 is to increase the already accelerated scrubbing period slightly, and to ensure that the actual sector error has a chance to be found during the subsequent scrubbing period. For example, if a disk is predicted to have a risk degree of 3, the scrubbing acceleration factor is 1.3, and the scrubbing rate controller will issue a scrubbing request to complete the scrubbing operation of this disk within 11.7 days (calculated by Equation 1). After completing the disk scrubbing, the controller will record the information about when the sector error occurred.

2) **Locate High-risk Areas at Sector Level***:* The probability that a sector error occurs in certain local areas of a disk is different from that in other areas. The question is how to locate the high-risk areas in a disk for which we have already observed historical sector errors. We apply four different disk models from company F[1] (one of the largest social network companies in the world), and explore the cumulative sector error distribution probability in different continuous high-risk areas (**sectors** around the failed sector, as illustrated in Figure 4). Note

[1]Note that we anonymize the company's name due to the double blind review policy.

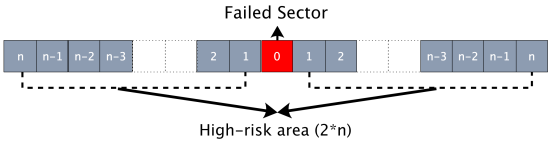


Fig. 4. Illustration of a high-risk area in a disk with a failed sector. The size of a high-risk area denotes the number of sectors around the failed sector in one disk.

that all disks we used in this experiment have historical sector errors, and that the sector mentioned in our paper is usually a physical sector of 4KB as in a real world data center. The results of our experiments are plotted in Figure 5. The x-axis shows the size of the high-risk area and the y-axis denotes the cumulative sector error probability. For most disk models, the probability that a new sector error occurs near an existing error within a high-risk area of size 10^7 is about 80%. We denote this phenomenon as *locality of sector errors*. In other words, the new sector error is correlated with existing errors, and there is a high probability that it appears in the vicinity of an existing error in the same disk. We set the size of the high-risk area to 10^7 sectors (as explained in Section III-B2).

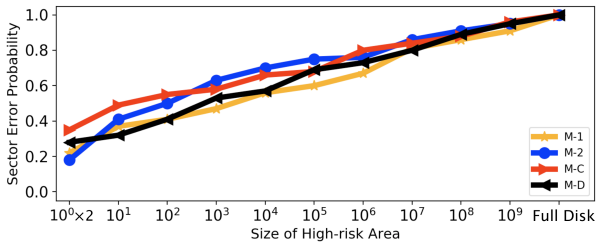


Fig. 5. The cumulative probability that a new sector error occurs near an existing error within differently sized high-risk areas. M-1, M-2, M-C and M-D represent four different disk models from company F.

3) **Piggyback Scrubbing at the Level of I/O Operations***:* The next question we address is how to opportunistically optimize the scrubbing strategy to accelerate the scrubbing in high-risk areas of an LSE disk. Part one on the top of Figure 6 illustrates examples of our piggyback scrubbing strategy, while part two on the bottom shows the differences to the sequential scrubbing strategy. We refer to the time of an I/O request as `I/O_TIME`, the time between I/Os requests as `I/O_INTERVAL` and denote the remaining untouched sectors as `FRAGMENTED_SECTOR`. After application I/O operations access the high-risk areas, the disk head returns to the scrubbing area to scrub, under a sequential scrubbing strategy. In contrast, once the application I/O operations access the high-risk areas, our scheme immediately scrubs the fragmented sectors untouched by the I/O operations, and only then resumes sequential scrubbing. This method, which can scrub the sectors in high-risk areas in advance coupled with application I/O operations, is refered to by us as *piggyback scrubbing strategy*. Furthermore, in this case, the disk head just needs to seek within a tiny area after conducting I/O operations. It is worthy to note that we perform sequential scrubbing with an adaptive accelerated rate (the scheme SU uses the sequential scrubbing with a fixed accelerated rate) on the entire LSE disk, and that the piggyback scrubbing strategy is only executed when application I/O operations access the high-risk areas. Although the piggyback operation will incur a certain scrubbing cost, it will ultimately reduce the cost of sequential scrubbing because we reduce the frequency of head movement for the fragmented sectors, which will have been scrubbed

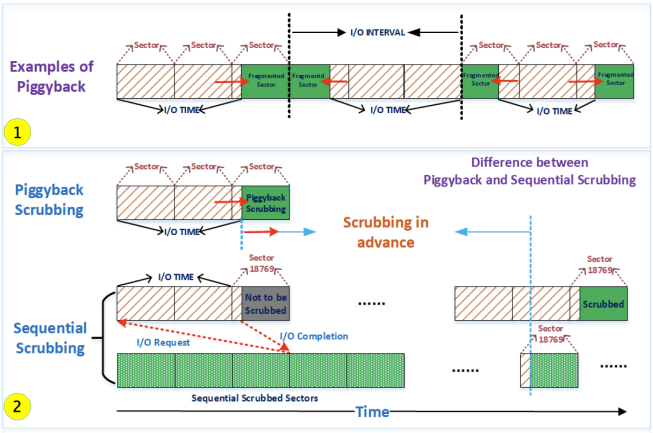


Fig. 6. Part one on the top describes gives an example of piggyback scrubbing; part two on the bottom illustrates the difference between piggyback and sequential scrubbing. Piggyback scrubbing strategy scrubs sectors in high-risk areas in advance, and reduces the frequency of head movement compared to the sequential scrubbing strategy.

by our piggyback strategy. Therefore, our strategy can not only scrub the high-risk areas earlier (with a big impact on the MTTD), but will also decrease the overall scrubbing cost (see Section III-B2 for details).

For a fair comparison, we use the same evaluation metrics as the scrub unleveling scheme SU from previous work [1]. The main idea of this method is to accelerate the scrubbing rate by a fixed factor $X(>1)$ if a sector error is predicted, and otherwise decelerate the scrubbing rate by a factor $Y(<1)$. We refer to the resulting MTTD and scrubbing cost of SU as $MTTD_{su}$ and $Cost_{su}$. The metrics are calculated as follows:

$$MTTD_{su} = \frac{\frac{1}{X2r}TP_{su} + \frac{1}{Y2r}FN_{su}}{P} \quad (2)$$

$$Cost_{su} = T(PP_{su}Xr + PN_{su}Yr) \quad (3)$$

where r is the rate at which a drive is being scrubbed, T is the time span of sequential scrubbing in the scheme SU, TP (FP) is the number of disks correctly (and incorrectly) predicted as having LSEs, TN (FN) is the number of disks correctly (and incorrectly) predicted as not having LSEs, with $P = TP+FN$, $PP = TP+FP$, $PN = FN + TN$. In our scheme TS, we adaptively accelerate the scrubbing rate by the adaptive factor X_a for all the LSE disks, after the sector risk degree prediction. Next, we verify disks that have historical sector errors by executing the piggyback scrubbing strategy in the high-risk areas. We calculate the MTTD and scrubbing cost of our scheme TS (refered to as $MTTD_{ts}$ and $Cost_{ts}$) as follows:

$$\begin{aligned} MTTD_{ts} &= \frac{(\frac{1}{X_a2r} - t)\delta TP_{h(ts)} + \frac{(1-\delta)TP_{h(ts)}}{X_a2r} + \frac{TP_{nh(ts)}}{X_a2r} + \frac{1}{Y2r}FN_{ts}}{P} \\ &= \frac{\frac{1}{X_a2r}TP_{ts} + \frac{1}{Y2r}FN_{ts} - t\delta TP_{h(ts)}}{P} \end{aligned} \quad (4)$$

$$Cost_{ts} = T'(FP_{ts}X_ar + PN_{ts}Yr) + (T' + \Delta T)TP_{ts}X_ar \quad (5)$$

where t is the scrubbing time in advance based on our piggyback scrubbing strategy. $TP_{h(ts)}$ $(TP_{nh(ts)})$ are the numbers of disks which are correctly predicted as LSE ones with (without) historical sector errors, where $TP_{ts} = TP_{h(ts)} + TP_{nh(ts)}$. δ is the proportion of the high-risk areas of the entire disk and ΔT is the additional scrubbing time incurred by piggybacking. Note that all these numbers are related to the high-risk areas discussed in Section II-B2. The larger the high-risk areas are, the larger δ and ΔT will be. Note

that the T' in our scheme is smaller than the T in Equation 3 because our piggyback scrubbing strategy reduces the frequency of head movement in sequential scrubbing compared to the scheme SU.

III. EXPERIMENTAL EVALUATION

In this section, we evaluate our scheme TS against the SU scheme with respect to four evaluation metrics.

A. Methodology

1) Datasets: We use S.M.A.R.T datasets from two real world data centers for evaluation. One is the publicly available dataset from 'Backblaze' [2] (we use the same disk models as [1] for a fair comparison), which spans a period of 50 months. The second proprietary dataset has been collected by company F and spans 26 months. Table I gives an overview of the summary statistics of these two datasets. Analogous to [11], we declare a sector error to occur when the raw value of the 5th S.M.A.R.T attribute "Reallocated Sectors Count" increases (which indicates the total number of reallocated sectors). We conduct experiments using three real world workloads from company F and denote them as W-A, W-B and W-C.

TABLE I
SUMMARY STATISTICS OF THE TWO EVALUATION DATASETS

Data center	Model	State	NO. Drives	NO. Samples
Backblaze	ST4000DM000(M-A)	Non-LSEs	37,006	12,433,874
		LSEs	396	9,258
	ST8000DM002(M-B)	Non-LSEs	11,253	3,557,212
		LSEs	338	9,808
F	WDC-A(M-C)	Non-LSEs	5,742	96,490,231
		LSEs	165	2,668,800
	WDC-B(M-D)	Non-LSEs	12,932	231,555,830
		LSEs	419	8,676,916

TABLE II
THE SMART ATTRIBUTES FOR OUR EVALUATIONS

#ID	S.M.A.R.T Attribute Name	Attribute type
001	Raw Read Error Rate	Normalized
003	Spin-Up Time	Normalized
004	Start/Stop Count	Raw
005	Reallocated Sectors Count	Raw
007	Seek Error Rate	Normalized
009	Power-On Hours	Normalized
010	Spin Retry Count	Normalized
012	Power Cycle Count	Raw
187	Reported Uncorrectable Errors	Normalized
194	Temperature Celsius	Normalized
197	Current Pending Sector Count	Raw
198	Offline Uncorrectable Sector Count	Raw

2) Experiment Setup: As shown in Table I, the original datasets contain more samples of non-LSE disks than of LSE disks, which is a situation referred to as an imbalanced dataset in the ML field. We apply a common technique called majority class under-sampling [23] to under-sample the majority class, which results in different ratios of LSE to non-LSE samples ranging from 1:1 to 1:50. We split the datasets into 70% training data and 30% testing data for our experiments, analogous to [1]. Furthermore, we obtain all results via cross-validation [24] to decrease the variability of the predictions. Note that our method uses the same S.M.A.R.T attributes as shown in Table II. Since different S.M.A.R.T attributes have different output ranges, (which might lead to different impacts on the predictive model), we normalize the range of all S.M.A.R.T attributes using min-max normalization, a common preprocessing technology in ML: $x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$, where x is the original value of a S.M.A.R.T attribute, x_{max} and x_{min} are the maximum and minimum value of the attribute in the training set, respectively. Note that we tried other normalization methods (e.g., z-score), but achieved the best results with min-max normalization.

[2]https://www.backblaze.com/b2/hard-drive-test-data.html

3) Evaluation Metrics: We use four metrics to report the results. **AUC (Area under the receiver operating characteristic curve)** is commonly used metric for evaluating a binary classification model. **Accuracy** is commonly used for evaluating a multiclass classification model. **MTTD** and **scrubbing cost** are used for estimating the reliability and efficient of our scheme.

AUC. We use the AUC value under the ROC curve (receiver operating characteristic) to evaluate the binary classification performance (to distinguish LSE disks from non-LSE disks) of our predictor in the ASRC. ROC is a curve plotting the False Detection Rate (FDR, also called recall rate) against the False Positive Rate (FPR) where FDR is on the y-axis and FPR is on the x-axis. FDR is the proportion of LSE disks that are correctly predicted, while FPR is the proportion of non-LSE disks that are falsely predicted as LSE disks. AUC is the area under this curve. Therefore, a higher the AUC means the model is better at distinguishing LSE disks and non-LSE disks.

Accuracy. We use the metric classification accuracy to evaluate the multiclass classification performance (identifying the sector risk degree of an LSE disk) of the predictor in the ASRC. It captures the proportion of the sector risk degrees that are correctly predicted. The higher the accuracy is, the better the predictor is.

Improvement in MTTD & Cost. We measure the ratio of improvement in MTTD and scrubbing cost as follows:

$$M_r = \frac{MTTD_{ts}}{MTTD_{su}} \quad \textbf{(The lower, the better)} \tag{6}$$

$$C_r = \frac{Cost_{ts}}{Cost_{su}} \quad \textbf{(The lower, the better)} \tag{7}$$

where M_r is related to the predictive performance of the ASRC and δ, and the ratio of improvement in scrubbing cost C_r is related to the predictive performance of the ASRC and ΔT, which we can conclude from Equation 2 to 5.

B. Experimental Results

In this section, we first show the predictive results of ASRC, and then analyze the MTTD and scrubbing cost with different sizes of high-risk areas respectively.

TABLE III
THE RESULTS OF AUC & ACCURACY. OUR PREDICTOR ACHIEVES BETTER PREDICTION QUALITY IN ALL CASES (BINARY CLASSIFICATION (HIGHER AUC) AS WELL AS MULTICLASS CLASSIFICATION (HIGHER ACCURACY))

Data center	Model	Method	Metric	Predictive Results
Backblaze	M-A	TS / SU	AUC	**0.92** / 0.87
		TS	Accuracy (0)	**97.6%**
			Accuracy (1-7)	**93.3%**
	M-B	TS / SU	AUC	**0.86** / 0.79
		TS	Accuracy (0)	**94.4%**
			Accuracy (1-7)	**92.1%**
F	M-A	TS / SU	AUC	**0.77** / 0.65
		TS	Accuracy (0)	**90.7%**
			Accuracy (1-7)	**89.9%**
	M-B	TS / SU	AUC	**0.75** / 0.64
		TS	Accuracy (0)	**88.6%**
			Accuracy (1-7)	**85.8%**

1) ASRC: Next, we conduct experiments to investigate the AUC of TS and SU as well as the predictive accuracy at different risk degrees using four disk models from two real world data centers. The results in Table III show that our predictor achieves a higher AUC than SU in all cases. We attribute this to the fact that our LSTM-based predictor

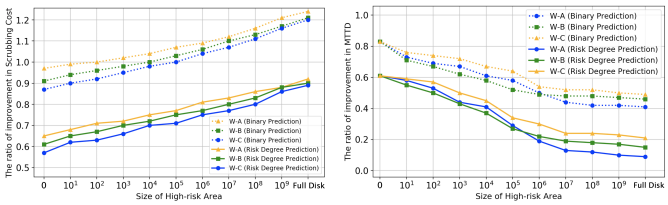
benefits from modeling the temporal and sequential dependencies. Due to the class imbalance between the non-LSE (degree 0) disks and LSE (degree 1-7) disks, we report the accuracy separately in Table III. Our predictor also achieves good predictive performance at different risk degrees which is important for determining an adaptive disk scrubbing rate at the disk level.

2) Scrubbing Cost and MTTD: Figures 7(a) & 7(b) show the ratio of improvement in scrubbing cost and MTTD under different sizes of high-risk areas using three real world workloads W-A, W-B and W-C. The only difference between the dotted lines and the solid lines is that the results of the dotted lines only use the binary classification results for fixed rate scrubbing acceleration, while the solid lines use the multiple classification results for adaptive rate scrubbing acceleration. In general, our scheme achieves lower MTTD and scrubbing cost than the scheme SU in all cases (C_r and M_r are all smaller than 1 in the solid lines). Moreover, the results of the solid lines achieve better performance than the dotted ones, which demonstrates that the predictor we designed in ASRC for the sector risk degree is more efficient than just predicting whether the disk is an LSE disk or not.

We briefly summarize additional findings for certain settings from these two figures. **(1) Not applying the piggyback scrubbing strategy.** Setting the high-risk area size to 0 means only using the predictor's results to accelerate scrubbing without using our piggyback scrubbing strategy. In that case, the ratio of the improvement in scrubbing cost and MTTD of our scheme only uses the binary predictive results (the dotted lines when setting the size of the high-risk area to 0) also achieves a lower scrubbing cost and a lower MTTD compared to the predictor in SU. The reason is that the LSTM-based predictor in our ASRC achieves higher binary classification performance (higher AUC) than the machine learning method the scheme SU adopted without considering the sequential dependency in the S.M.A.R.T training data. **(2) Applying the piggyback scrubbing strategy to the full disk area.** Setting the high-risk area size to a full disk means that we do not use the locality of sector errors to focus on high-risk areas. Although we achieve the smallest $MTTD_{ts}$ (the lowest M_r) in this case, we incur unnecessary additional scrubbing cost because the $MTTD_{ts}$ does not change a lot compared to the results with an area size of 10^7. This demonstrates the effectiveness of exploiting the locality of sector errors. It is worthy to note that even if we execute the piggyback strategy in the full disk area, which incurs the highest scrubbing cost, our scheme results in a lower scrubbing cost than SU (C_r = 0.9, which is smaller than 1 in the solid lines when setting the size of the high-risk area to the full disk). We attribute this to the fact that the piggyback scrubbing also reduces the cost of the sequential scrubbing strategy (as discussed in detail in SectionII-B3). **(3) The configuration of our real world large scale storage system.** Limiting the high-risk area size to 10^7 sectors can simultaneously decreases the MTTD by about 80% (M_r = 0.2) and the scrubbing cost (C_r = 0.8) by about 20%, compared to the scheme SU.

IV. CONCLUSION

In this paper, we proposed an adaptive and tiered disk scrubbing scheme with improved MTTD and reduced scrubbing cost. Our main contributions include: (1) we are the first to propose an Adaptive Scrubbing Rate Controller (ASRC) to not only predict LSE disks and non-LSE disks but also the sector risk degree to accelerate the disk scrubbing at an adaptive (not fixed) rate based on the LSTM algorithm at disk level, (2) we use the locality of sector errors to focus on high-risk areas of the LSE disks at sector level, and (3) we propose a piggyback scrubbing strategy to optimize the scrubbing strategy



(a) C_r **(The lower, the better)** (b) M_r **(The lower, the better)**

Fig. 7. The x-axis in these two figures shows the size of the high-risk areas, the y-axis shows the ratio of improvement in scrubbing cost and MTTD. Our scheme TS achieves lower MTTD and scrubbing cost than SU in all cases (C_r and M_r are all smaller than 1 in the solid lines).

to further achieve higher reliability and lower scrubbing cost which has important practical applicability in the realistic large scale data centers. Our experiments on datasets and workloads from two real world data centers have shown that we decrease about 80% MTTD while decreasing 20% scrubbing cost compare to the scheme SU [1].

REFERENCES

[1] T. Jiang and P. H. et. al, "Scrub unleveling: Achieving high data reliability at low scrubbing cost," in *DATE*, 2019, pp. 1403–1408.
[2] C. B. et. al, "Windows azure storage: a highly available cloud storage service with strong consistency," in *SOSP*, 2011, pp. 143–157.
[3] C. H. et. al, "Erasure coding in windows azure storage," in *ATC*, 2012, pp. 15–26.
[4] B. R. et. al, "A case for redundant arrays of inexpensive disks (raid)," *SIGMOD*, pp. 109–116, 1988.
[5] J. Z. et al., "Transfer learning based failure prediction for minority disks in large data centers of heterogeneous disk systems," in *ICPP*, 2019, pp. 66:1–66:10.
[6] Z. B. et. al, "Proactive drive failure prediction for large scale storage systems," in *MSST*, 2013, pp. 1–5.
[7] S. Xiaoyi and C. et. al, "System-level hardware failure prediction using deep learning," in *DAC*, 2019, pp. 20:1–20:6.
[8] P. et. al, "A comparison of machine learning algorithms for proactive hard disk drive failure detection," in *ISARCS*, 2013, pp. 1–10.
[9] Y. W. et. al, "Hard drive failure prediction using big data," in *SRDSW*, 2015, pp. 13–18.
[10] A. Bruce, *Monitoring hard disks with smart*, 2004, no. 117.
[11] F. M. et. al, "Improving storage system reliability with proactive error prediction," in *ATC)*, 2017, pp. 391–402.
[12] S. V. et. al, "Random forest: a classification and regression tool for compound classification and qsar modeling," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 6, p. 1947, 2003.
[13] Z. Y. et. al, "Predicting disk failures with hmm- and hsmm-based approaches," 2010, pp. 390–404.
[14] S. et. al, "Lstm neural networks for language modeling," in *INTERSPEECH*, 2012.
[15] H. et. al, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[16] C. C. et. al, "Support-vector networks," *JMLR*, vol. 20, no. 3, pp. 273–297, 1995.
[17] J. R. Quinlan, "Induction on decision tree," *JMLR*, vol. 1, no. 1, pp. 81–106, 1986.
[18] H. D. W. et. al, *Introduction to the Logistic Regression Model*, 2005.
[19] H. W. et. al, "Action Recognition by Dense Trajectories," in *CVPR*, 2011, p. 3169–3176.
[20] P. et. al, "Cross-language text classification using structural correspondence learning," in *ACL*, 2010, pp. 1118–1127.
[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[22] A. Oprea and A. Juels, "A clean-slate look at disk scrubbing," in *FAST*, 2010.
[23] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE TKDE*, vol. 21, no. 9, pp. 1263–1284, 2009.
[24] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, 2000.