Cluster storage systems gotta have HeART: improving storage efficiency by exploiting disk-reliability heterogeneity

Saurabh Kadekodi, K. V. Rashmi, and Gregory R. Ganger, ***Carnegie Mellon University***

https://www.usenix.org/conference/fast19/presentation/kadekodi

This paper is included in the Proceedings of the 17th USENIX Conference on File and Storage Technologies (FAST '19).

February 25–28, 2019 • Boston, MA, USA

978-1-939133-09-0

Open access to the Proceedings of the 17th USENIX Conference on File and Storage Technologies (FAST '19) is sponsored by

NetApp®

Cluster storage systems gotta have HeART: improving storage efficiency by exploiting disk-reliability heterogeneity

Saurabh Kadekodi, K. V. Rashmi, Gregory R. Ganger
Carnegie Mellon University

Abstract

Large-scale cluster storage systems typically consist of a heterogeneous mix of storage devices with significantly varying failure rates. Despite such differences among devices, redundancy settings are generally configured in a one-scheme-for-all fashion. In this paper, we make a case for exploiting reliability heterogeneity to tailor redundancy settings to different device groups. We present HeART, an online tuning tool that guides selection of, and transitions between redundancy settings for long-term data reliability, based on observed reliability properties of each disk group. By processing disk failure data over time, HeART identifies the boundaries and steady-state failure rate for each deployed disk group (e.g., by make/model). Using this information, HeART suggests the most space-efficient redundancy option allowed that will achieve the specified target data reliability. Analysis of longitudinal failure data for a large production storage cluster shows the robustness of HeART's failure-rate determination algorithms. The same analysis shows that a storage system guided by HeART could provide target data reliability levels with fewer disks than one-scheme-for-all approaches: 11–16% fewer compared to erasure codes like 10-of-14 or 6-of-9 and 33% fewer compared to 3-way replication.

1 Introduction

Large cluster storage systems almost always include a heterogeneous mix of storage devices, even when using devices that are all of the same type (e.g., Flash SSDs or mechanical HDDs). Commonly, this heterogeneity arises from incremental deployment combined with per-acquisition optimization of which make/model to acquire, such as targeting the lowest cost-per-byte option available at the time. As a result, a given cluster storage system can easily include several makes/models, each in substantial quantity.

Beyond performance and capacity differences, different makes/models can also have substantially different reliabilities. For example, Fig. 1 shows the average *annualized failure rates (AFRs)* during the useful life (stable operation period) for the 6 HDD make/model-based disk groups that make up more than 90% of the cluster storage system (with 100,000+ disks) used for the Backblaze backup service [5]. The highest failure rate is over 3.5× greater than the lowest, and no two are the same. Schroeder et al. [32] recently showed that different Flash SSD makes/models similarly exhibit substantial failure rate differences.

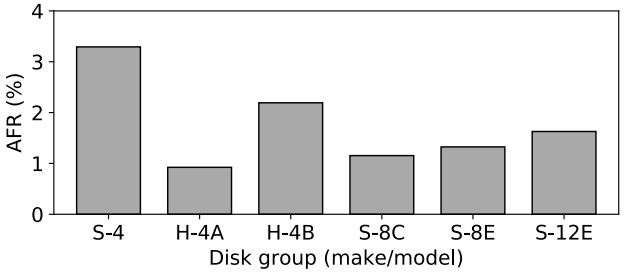


Figure 1: Annualized failure rate (*AFR*) for the six disk groups that make up >90% of the 100,000+ HDDs used for the Backblaze backup service [5]. Details of each disk group are given in Section 2.

Despite such differences, the degree of redundancy employed in cluster storage systems for the purpose of long term data reliability (e.g., the degree of replication or erasure code parameters) is generally configured as if all of the devices have the same reliability. Unfortunately, this approach leads to configurations that are overly resource-consuming, overly risky, or a mix of the two. For example, if the redundancy settings are configured to achieve a given data reliability target (e.g., a specific *mean time to data loss (MTTDL)*) based on the highest *AFR* of any device make/model (e.g., S-4 from Fig. 1), then too much space will be used for redundancy associated with data that is stored fully on lower *AFR* makes/models (e.g., H-4A). Continuing this example, our evaluations show that the overall wasted capacity can be up to 16% compared to uniform use of erasure code settings stated as being used in real large-scale storage clusters [13, 25, 26, 28] and up to 33% compared to using 3-replication for all data—the direct consequence is increased cost, as more disks are needed. If redundancy settings for

all data are based on lower *AFR*s, on the other hand, then data stored fully on higher-*AFR* devices is not sufficiently protected to achieve the data reliability target.

This paper presents HeART (Heterogeneity-Aware Redundancy Tuner), an online tool for guiding exploitation of reliability heterogeneity among disks to reduce the space overhead (and hence the cost) of data reliability. HeART uses failure data observed over time to empirically quantify each disk group's reliability characteristics and determine minimum-capacity redundancy settings that achieve specified target data reliability levels. For the Backblaze dataset of 100,000+ HDDs over 5 years, our analysis shows that using HeART's settings could achieve data reliability targets with 11–33% fewer HDDs, depending on the baseline one-scheme-for-all settings. Even when the baseline scheme is a 10-of-14 erasure code whose space-overhead is already low, HeART further reduces disk space used by up to 14%.

Online (real-time) use of observed device reliability requires careful design. HeART uses robust statistical approaches to identify not only a steady-state *AFR* estimate for each disk group, but also the transitions between deployment stages: infancy→useful life→wearout, as in bathtub curve visualizations. HeART assumes that administrators have a baseline redundancy configuration that would be used in HeART's absence; that same configuration should be used for a disk group, when it is initially deployed. HeART processes failure data for that disk group, during this initial period of 3–5 months, to determine both when infancy ends and a conservative *AFR* estimate for the useful life period. It also suggests the most space-efficient redundancy settings supported by the storage system that will achieve the specified data reliability target.

Naturally, the useful life period does not last forever. HeART continues to process failure data for each disk group, automatically identifying the onset of the wearout period. At this point, a transition to more conservative redundancy (e.g., the original baseline), and possibly decommisioning, is warranted. Importantly, HeART distinguishes between anomalous failure occurrences (e.g., one-time device-independent events, like a power surge, in which many devices fail together) and true changes in the underlying *AFR*.

This paper makes four primary contributions. First, it highlights an often overlooked aspect of device heterogeneity (reliability) that should be exploited in cluster storage systems, and quantifies potential cost–and/or–reliability benefits. Second, it confirms the above observation and quantification with analysis of multi-year reliability data from a sizable cluster storage deployment (Backblaze), showing up to 11–33% reduction in the overall number of disks needed to achieve target data reliability. Third, it describes an online tool (HeART) that automatically determines per-disk-group useful life *AFR*s and durations, and identifies the right redundancy scheme settings for each. Fourth, it shows that HeART's algorithms are effective using data from a large-scale production cluster (Backblaze) and are able to expose the expected capacity savings opportunities without compromising data reliability.

Make/Model	Disk group shorthand	# of disks	Oldest disk age
Seagate ST4000DM000	S-4	37015	5 yrs
HGST HMS5C4040ALE640	H-4A	8715	4.77 yrs
HGST HMS5C4040BLE640	H-4B	15048	4.2 yrs
Seagate ST8000DM002	S-8C	9885	1.99 yrs
Seagate ST8000NM0055	S-8E	14395	1.2 yrs
Seagate ST12000NM0007	S-12E	21581	8 mts

Table 1: The disk groups identified from the Backblaze dataset for reliability heterogeneity analysis. The disk group shorthand above is used to represent the respective makes/models throughout the paper.

2 Having HeART can make you rich

This section builds a case for HeART by showing the benefits of using different redundancy schemes for disk groups exhibiting different reliability characteristics in the same commercially used cluster storage system. To support the case, we quantify space overhead reductions that can be achieved by adopting the different redundancy schemes.

2.1 The Backblaze dataset

Our analysis is based on an open source dataset from a data backup organization, Backblaze [5]. This dataset consists of over 5 years of disk reliability statistics from a production cluster storage system with over 100,000 HDDs.

We use the standard metric, *annualized failure rate (AFR)*, to describe a disk's fail-stop rate[1] [9, 33]. As the name suggests, it is the expected percentage of disks that will fail-stop in a given year from a population of disks. *AFR* is calculated on day d, based on the past d days of reliability data, using the following formula:

$$AFR\ (\%) = \frac{f_d}{n_1 + n_2 + \ldots + n_d} \times 365 \times 100 \qquad (1)$$

where f_d is the number of disks failed in the past d days and n_i is the number of disks operational during day i.

Note that the AFR calculation is dependent on the number of days a disk was in operation. This can be tricky to estimate from the Backblaze dataset since the "death" of a disk in this dataset may also indicate its decommissioning, which may or may not imply its failure. We argue that, in the case of Backblaze, the date of decommissioning a disk only affects the absolute date at which it would have fail-stopped, but does not affect its *rate of failure*. Backblaze adopts a proactive disk replacement strategy that is driven by monitoring a combination of five S.M.A.R.T. (Self-Monitoring, Anal-

[1] Storage devices can exhibit *partial failures* and *fail-stops (complete failures)*. Partial failures might involve a particular read or write failing because of a sector error or checksum failure, while the disk as a whole is still functional. In the case of fail-stop, the disk stops functioning altogether.

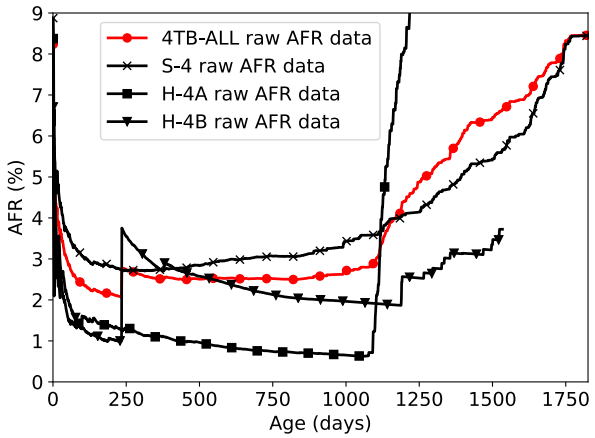


Figure 2: AFR comparison between all 4TB disks grouped together and disk groups broken down by make/model. The *AFR* differences in make/model-based grouping enables HeART to perform finer-grained specialization leading to higher benefits.

ysis and Reporting Technology) statistics.[2] The increased probability of failure indicated by grown defects in a disk is supported by several previous studies [7, 20, 24, 29]. In fact, Pinheiro et al. [24] show that the critical threshold for several S.M.A.R.T. attributes before their imminent failure is one—that is, the probability of failure of a disk in the next two months increases manifold when any of these S.M.A.R.T. attributes show a value greater than zero. Ma et al. [20] also show the high likelihood of disk failure by monitoring the reallocated sectors count (S.M.A.R.T. attribute 5), which is one of the signals used by Backblaze as a disk replacement indicator. Therefore, we believe that Backblaze's proactive disk replacement rate is a reasonable approximation for the actual disk failure rate.

2.2 Disk group formation and varying *AFR*s

To effectively exploit heterogeneity in *AFR*s of different disk groups, we need to categorize the disks using some parameter that (1) groups disks with similar *AFR*s together and (2) has substantially different *AFR*s across groups. In whichever manner we choose to group the disks, in order to gain statistical confidence in the *AFR* value, we need to ensure that each disk group has a sizeable population. Our definition of a sizeable population is approximately 10,000 or more disks. This is in line with disk populations considered in previous reliability studies [21]. We identify the following four ways to categorize disks:

- **By make/model:** Economies of scale result in large quantities of disks being purchased from the same vendor. Prior studies have shown that *AFR* may vary significantly by vintage [10, 20, 24].
- **By capacity:** Grown defects can be a function of disk capacity, thus causing disks of similar capacity to fail at a similar rate.
- **By operational conditions:** Disks that share similar vibration or temperature experiences may cause them to fail similarly. Thus, chassis placement and other operational conditions may influence failure rates.
- **By usage:** Increased space utilization or higher I/O rates may result in different disks showing different failure characteristics.

[2] Backblaze uses S.M.A.R.T. 5 (Reallocated Sectors Count), S.M.A.R.T. 187 (Reported Uncorrectable Errors), S.M.A.R.T. 188 (Command Timeout), S.M.A.R.T. 197 (Current Pending Sector Count) and S.M.A.R.T. 198 (Uncorrectable Sector Count) as indicators that a disk is about to fail. [6]

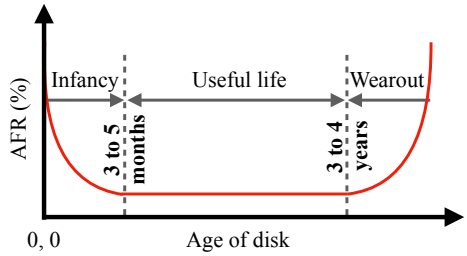


Figure 3: The canonical bathtub curve used to represent disk failure characteristics.

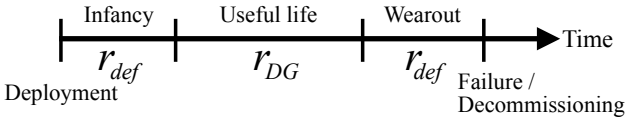


Figure 4: An abstract timeline of a disk group from deployment to failure or decommissioning, with the three distinct periods. The notations below the timeline (r_{def} and r_{DG}) denote the redundancy scheme employed during the respective stage.

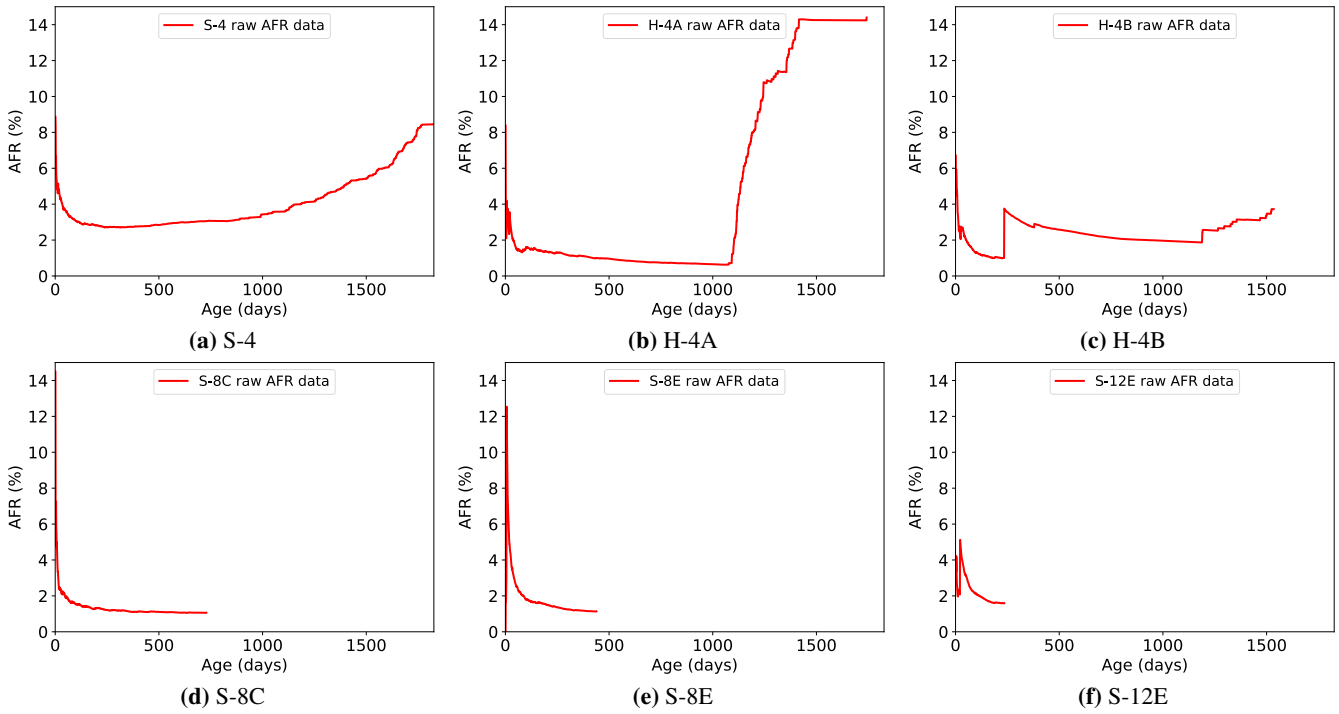
Unfortunately since we do not have access to the operational conditions or usage patterns, we can only analyze grouping on the basis of make/model or capacity.

Fig. 2 shows the *AFR* by considering all 4TB disks as one disk group (red curve with circular marks) and the *AFR*s of the three make/models of 4TB disks as individual disk groups (black curves). We see significant differences between *AFR*s when disks are grouped by make/model, suggesting that grouping by capacity is insufficient. HeART groups disks by make/model.

Table 1 shows the six make/model disk groups that make up over 90% of the Backblaze deployment, with their population size, the age of their oldest disk, and the shorthand names we will use throughout the paper.

***AFR* variation over time.** As expected, *AFR* values of each disk group vary over the lifetime of disks. It is well known that the *AFR* values over a disk's lifetime follow a *bathtub curve* [11, 12, 45]. Fig. 3 shows the canonical representation of a bathtub curve. The lifetime is typically divided into three distinct periods:

- **Infant mortality:** A higher failure rate in the early days after deployment. This is also called the *burn-in period*.
- **Useful life period:** The stable region of operation, where rate of failure is lower.
- **Wearout stage:** A higher failure rate towards the disks' end of life due to wear and aging.



(a) S-4 **(b)** H-4A **(c)** H-4B

(d) S-8C **(e)** S-8E **(f)** S-12E

Figure 5: Cumulative raw *AFR* versus age (in days) for all six disk groups being analyzed.

Fig. 5 shows the *AFR* behavior versus age of the six disk makes/models. The three disks in the top row clearly exhibit all three stages of the bathtub curve.[3] This is because the oldest disks from the S-4, H-4A and H-4B disk groups in the dataset are old enough to have entered their wearout stages. Since the deployment of S-8C, S-8E and S-12E disks has been more recent, these disk groups have ended their infant mortality, but are yet to enter their wearout stages.

2.3 Space savings from heterogeneous *AFR*s

Our goal is to reduce storage overhead by tailoring the redundancy scheme employed to the failure rate of a disk group during its useful life period. We parameterize a redundancy scheme using two parameters n (called "length") and k (called "dimension"), and call it a (n,k) scheme.[4] For any replication based scheme, $k = 1$ and n represents the total number of replicas. For any erasure coding based scheme, k represents the number of data chunks and $(n-k)$ is the number of parity chunks, thus resulting in n chunks in total.[5] For an (n,k) redundancy scheme, the storage overhead is given by $\frac{n}{k}$.

HeART achieves reduction in storage overhead by explicitly factoring in the group-specific *AFR* values in deciding the appropriate redundancy scheme for each disk group. Based on the canonical bathtub curve (Fig. 3), and the *AFR* curves shown in Fig. 5, we conclude that the safest stage to apply lower redundancy (without risking not meeting the reliability target) during a disk group's lifetime is in its useful life period. Fig. 4 shows the abstract timeline of a disk group, where r denotes the redundancy scheme applied in each stage. Since all cluster storage systems today use some redundancy scheme whose resilience is acceptable to them, we assume that to be the *default* redundancy scheme. Since infancy and wearout periods have higher and less stable *AFR*s compared to useful life, for every disk group, HeART employs the default redundancy scheme for all infancy and wearout periods. This is shown as r_{def} in Fig. 4.

HeART suggests lower redundancy than the default scheme only during the useful life period, during which *AFR* values are relatively stable. Data redistribution and issues related to data placement and scheme transitions are discussed in Section 5.

We use the standard metric for reliability of data employed in storage systems, *mean time to data loss* (*MTTDL*). *MTTDL* is calculated based on two rates – *mean time to failure* (*MTTF*) and *mean time to repair* (*MTTR*) [23, 38] *MTTF* is directly related to the disk's *AFR*. *MTTR* is the time it takes to reconstruct the lost data on the failed disk. Following prior work, we model the time to repair based on the time it takes to detect that a disk has failed (which is approximately 15 minutes) [13, 17]. We note that, by choosing the failure detection time as a proxy for the repair time, we are effectively choosing a lower bound on

[3] Fig. 5c corresponding to disk H-4B does not completely conform to the bathtub shape. We will discuss this case later in detail.

[4] This notation follows the standard notation employed in the coding theory literature.

[5] Although the description of the notation applies only to "systematic" codes, and most of the codes employed in storage systems are indeed systematic, HeART is applicable to storage systems employing non-systematic codes as well.

the repair time. Reliability differences between redundancy schemes are higher when repair times are higher, leading to even greater potential for space saving through HeART.

When a disk group enters its useful life period, HeART chooses a redundancy scheme (r_{DG}) that meets the following conditions:

1. is as reliable as r_{def}, i.e. $MTTDL^{r_{DG}} \geq MTTDL^{r_{def}}$
2. tolerates at least as many failures as r_{def}

According to condition 1 above, we need to set a target $MTTDL$ in order to compare the resilience of different redundancy schemes. Although prior studies have shown $MTTDL$ targets to be as low as 10,000 years [27], in order to ensure that we do not regress on reliability that disks in our dataset can currently offer, we set the target $MTTDL$ to be the $MTTDL$ of the default redundancy scheme applied on the disk group with the highest AFR. S-4 is the disk group with highest useful life AFR in the Backblaze dataset (refer Fig. 1). Therefore, for every default redundancy scheme, we will use S-4's $MTTDL$ for that scheme as the target $MTTDL$.

Multiple redundancy schemes can achieve the same or similar $MTTDL$ values. These schemes can differ in their dimension (k) or the number of parity chunks per stripe ($n-k$) or both. It is well known that, generally speaking, codes with a longer dimension can provide the same $MTTDL$ with lower space overhead compared to shorter codes. However, long codes consume significantly higher cluster bandwidth for reconstruction, since many more disks have to be accessed when performing reconstruction of failed data [17, 25, 26, 28]. The cluster bandwidth consumed during reconstruction is a major concern in erasure-coded storage systems. This has been highlighted in several works in the past [17, 25, 26, 28] and is consistent with our discussions with cluster storage system administrators. We, therefore, limit our cost reduction analysis to codes with at most $2\times$ the dimension (i.e., parameter k) of the default redundancy scheme.

Table 2 shows space savings for one disk group (H-4A) as an example. We will first highlight the space reduction when erasure coding schemes are used as the default, focusing on the $(14,10)$ and $(9,6)$ schemes known to have been used in large data centers [13, 25, 26, 28]. For $(14,10)$ as the default scheme, the $MTTDL$ difference between H-4A and S-4 disks is over $580\times$. Thus, we can choose a weaker redundancy scheme (a scheme with lower storage overhead $\frac{n}{k}$), so long as conditions 1 and 2 above are fulfilled. In fact, the high AFR differences allow us to use the longest allowed optimized code ($2\times$ the dimension of the default redundancy scheme) for H-4A disks, i.e. $(24,20)$ leading to a useful life space reduction of 14%. Similarly, when using $(9,6)$ as the default scheme, the $MTTDL$ difference between H-4A and S-4 is over $160\times$. This again allows us to choose the longest code for H-4A when $r_{def} = (9,6)$, i.e. $(15,12)$, providing a space reduction of 16%.

Disk groups		r_{def} **= (14, 10)**		r_{def} **= (9, 6)**		r_{def} **= (3, 1)**	
DG	AFR	r_{DG}	**Cost↓**	r_{DG}	**Cost↓**	r_{DG}	**Cost↓**
S-4	3.29%	(14, 10)	**NA**	(9, 6)	**NA**	(3, 1)	**NA**
H-4A	0.92%	(24, 20)	**14%**	(15, 12)	**16%**	(4, 2)	**33%**

Table 2: A sample of the estimated savings achievable through HeART. The space reductions obtained on H-4A disks by using redundancy schemes with lower storage overhead while meeting the reliability target set by applying the default redundancy scheme (r_{def}) on S-4 disks.

For r_{def} = 3-replication (recall that, under the $(n,\ k)$ notation introduced above, 3-replication is denoted as the $(3,1)$ erasure code), we can tune the redundancy on H-4A disks to $(4,2)$ to respect our $2\times$ default stripe dimension limit and still achieve an $MTTDL$ that is approximately $11\times$ that of S-4's $MTTDL$. Using a $(4,2)$ scheme leads to a 33% reduction in disk space.

Large internet services companies try very hard to minimize free space (as low as 5%, according to some administrators) in order to minimize capital and operating costs. We are told that space savings translate directly into reduced numbers of disks needed, and even modest space savings (e.g., 10%) would build a solid case for tailoring redundancy schemes to heterogeneous disk AFRs.

We note that much of the reduction in storage overhead arises from allowing codes up to $2\times$ in dimension (i.e., parameter k). However, simply employing an erasure code with twice the dimension for all data is not generally a suitable solution. First, the AFR for certain disk groups might be high enough to make codes with $2\times$ dimension not acceptable causing them to miss the target reliability. Second, and more broadly, the reconstruction overheads can be unacceptable. For popular codes employed in practice, the amount of cluster bandwidth required for reconstruction is proportional to $k\times AFR$, where k is the dimension of the code. The stable and lower AFR during a disk group's useful life period allows the I/O generated for reconstruction to be contained even if longer codes are employed, which is why HeART optimizes redundancy schemes *only* during a disk group's useful life. Using longer codes on data stored on disk groups in their infancy and wearout stages would exacerbate the cluster bandwidth consumption for reconstruction due to higher failure rates in these stages.

3 The ways of the HeART

This section describes the challenges, design and implementation of HeART. We also quantify the cost reductions achieved by HeART for the Backblaze dataset.

3.1 Challenges

There are several challenges in taking the idea presented in Section 2 to practice.

Challenge 1: Function online and be quick. In making our case for HeART, we made use of the complete fail-

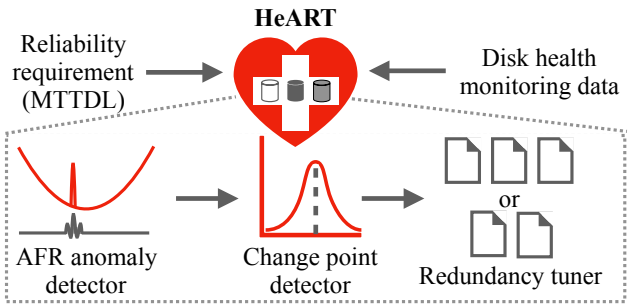


Figure 6: Schematic diagram of HeART. Components include an anomaly detector, an online change point detector, and a redundancy tuner.

ure information (e.g., the full bathtub curve) for the disk groups. This helped in clearly identifying the 3 stages of a disk group's lifetime and AFR values in each of the stages. In practice, however, AFR values for disk groups deployed in cluster storage systems can only be known in an online fashion (i.e., as a continuous stream of reliability data, as it is observed). Furthermore, the crux of the cost reduction from HeART comes from quickly tuning the redundancy scheme as soon as we are confident of a disk group having entered its useful life period. Thus, our first challenge in building HeART is that it needs to function in an online fashion taking a continuous stream of disk health data as input and quickly react to the changes in the failure rate.

Challenge 2: Be accurate. It is important to correctly identify the three different stages of the bathtub curve for each disk group (recall Fig. 3). If we are hasty in declaring the end of the infancy period or lax in identifying end of useful life, we might not meet the reliability target because of having tailored the redundancy to a relatively low failure rate during the useful life period. In contrast, if we are too lax about declaring end of infancy or too hasty in declaring onset of the wearout stage, the opportunity of cost reduction will diminish.

Challenge 3: Filter-out anomalies. Events such as power outages, natural disasters or human error can cause large numbers of disks to fail at once It is important to distinguish between an accidental rise in AFR due to such anomalous events versus the rise in AFR due to onset of the wearout stage. Our third challenge is to perform AFR anomaly detection to avoid prematurely declaring end of useful life, consequently reducing the window of opportunity for cost reduction. At the same time, HeART needs to exercise caution so as to not treat a genuine rise in AFR as an anomaly, which risks not meeting reliability targets.

3.2 HeART architecture

Fig. 6 shows the primary components of HeART. HeART assumes the existence of a disk health monitoring/logging mechanism already in place, which is common in large-scale cluster storage deployments. From the time of deployment till the end of infancy, the default redundancy scheme (r_{def}) is used to protect the data stored on a disk group. Periodically, disk health data for each disk group is passed through an *anomaly detector*. Following an anomaly check, the cumulative AFR of every disk group is passed through a *change point detector*, which checks if a transition to different phase of life has occurred. Once the change point detector announces start of the useful life period, HeART suggests a new redundancy mechanism for the useful life of the disk group (r_{DG}). It computes a *determined useful life AFR* (AFR_{DG}), which is the AFR at the end of infancy padded with a tunable buffer, and uses it to calculate $MTTDL^{r_{DG}}$ for different redundancy scheme (r_{DG}) options. The buffer is introduced to tolerate the fluctuation of AFR during the useful life period (see Section 4.3). HeART keeps checking for anomalies and change points throughout the useful life period. When the change point detector marks the end of useful life, HeART raises an alert to reset the redundancy scheme to r_{def} to handle the increased AFR during wearout, as was handled in the absence of HeART.[6]

The remainder of this section describes our approach to addressing the above mentioned challenges. We leverage established tools and algorithms from online services and time-series analysis literature. While other options may perform even better, our evaluations indicate that these established tools are effective. We show the efficacy of HeART using the Backblaze dataset in Section 4.

3.3 Online anomaly detection

Incidents like losing power to a rack of disks, a natural disaster, or an accident, can cause a large number of failures resulting in a sudden rise in AFR. Such bulk failures can easily exceed the limits of any reasonable redundancy scheme, so administrators seek to mitigate them by defining appropriate failure domains and spreading data+redundancy across the failure domains [25, 26]. Such failures are not reflective of the true rise in AFR because of wearout, and therefore HeART considers these incidents as anomalies. It is important to note that the benefits we extract from exploiting the reliability heterogeneity are proportional to the length of the useful life period, and therefore prematurely announcing wearout stage due to an anomaly would significantly diminish achievable gains.

We use the H-4B disks as a motivating example for anomaly detection (shown in Fig. 7). The raw AFR curve (red curve) shows that just after a few days into its useful life, there are large spikes in the AFR curve for drives that are about 235 days old (point A) and 380 days old (point B). Further along, we observe three more spikes that are in succession for disks that are about 1200 days old (points C, D and E). The failures corresponding to points A and B are all caused because of 322 drives failing on one particular date.

[6] We note that the current architecture of HeART determines one useful life AFR for all disks belonging to a disk group and does not handle changes in the intra-disk-group reliability distribution over time.

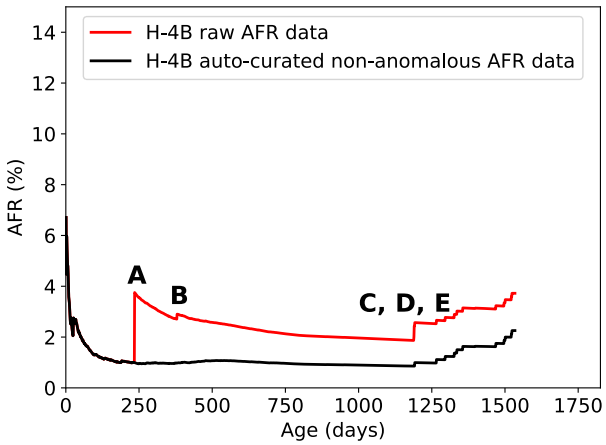


Figure 7: Raw and HeART-curated AFR curves for the H-4B disk group. Five spikes in *AFR* (points A–E), which correspond to four (anomalous) bulk failure events, are automatically filtered out by HeART.

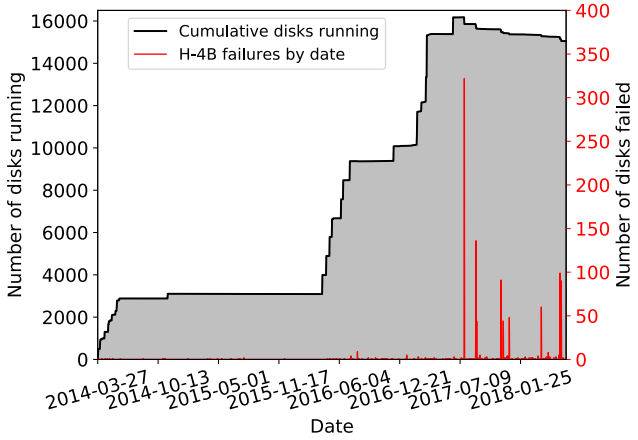


Figure 8: Total number of disks and number of disk failures by date for H-4B disks. The step-wise jumps in the black curve represent incremental deployments. The largest red spike represents the disks that failed on July 23, 2017, causing anomalies A and B in Fig. 7.

Here, failure of disks of two different ages correspond to a failure event on the same day because these disks were deployed on different dates. Fig. 8 shows the total number of disks running and the per-day number of disk failures of H-4B as a function of the date. The left y-axis shows the cumulative disks of H-4B running on each day. The steps in the black curve show the incremental deployments of H-4B disks. The right y-axis shows the number of H-4B disks failing on each day (red curve). The tallest red spike in Fig. 8 corresponds to points **A** and **B** from Fig. 7. Points C, D and E occurred because of disks failing on different days.

In the absence of anomaly detection, HeART would have incorrectly concluded that the disk group's wearout stage began as early as point A.

3.4 Online change point detection

We refer to a transition in the *AFR* curve of a disk group as a *change point*. There are two major change points for each disk group: end of infant mortality stage and the onset of the wearout stage. This subsection describes our methods of identifying the two change points.

Onset of useful life period. HeART uses prior studies about infant mortality in HDDs along with change point detection to decide a disk group's end of infancy. Prior studies performed on the Google and EMC disk fleets [20, 24] have shown that infant mortality lasts for approximately one quarter. Therefore, in order to be conservative, HeART exempts the first quarter from being assessed for end of infant mortality. Since disk reliability data is collected periodically, each time data is collected after the first 90 days, we run change point detection on the *AFR* curve generated by a sliding window of the past 30 days. HeART declares end of infancy if the last change point marked by the detector is over 30 days old, and the failure rate during the last 30 days is relatively constant. More precisely, HeART declares end of infancy when the difference between the observed maximum and minimum *AFR* values in at least 30 days past the last change point is less than a certain threshold T_{flat}. T_{flat} is the threshold for *flatness* and is a tunable parameter in HeART. Sensitivity to T_{flat} is evaluated in Section 4.3. Note that HeART takes a conservative approach in declaring the onset of the useful life period of a disk group in order to increase confidence about reducing redundancy for data stored on that disk group.

End of useful life period. Being lax in declaring the end of useful life period (i.e., onset of wearout) can risk in HeART not meeting the intended reliability target. Hence, HeART takes a conservative approach and marks the end of useful life for the first *AFR* observed that is greater than the determined useful life *AFR*. Since HeART checks for anomalous *AFR* fluctuations before checking for change points, if the anomaly detection phase does not filter out an increase in *AFR*, HeART assumes it to be a true increase in *AFR*. Thus, here too HeART takes a conservative approach and errs on the side of exiting the useful life period early and reverting to the default redundancy scheme.[7]

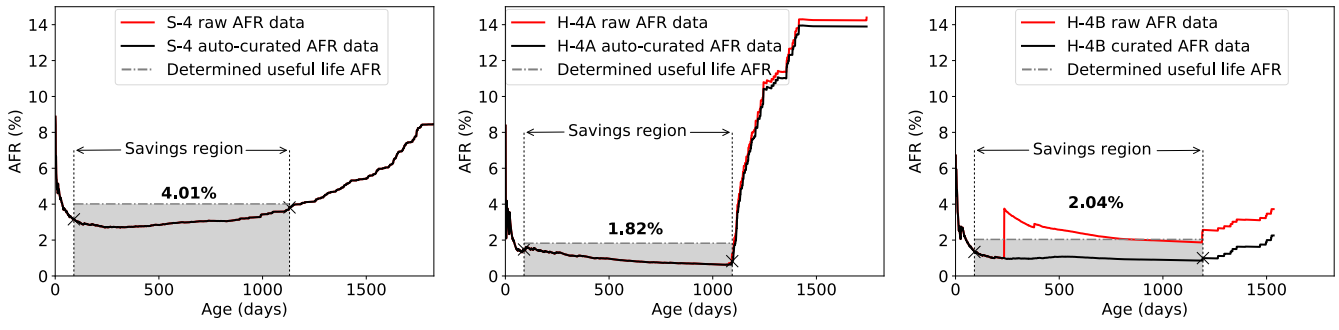
4 Measuring HeART

This section describes implementation details of various components that make up HeART and presents an evaluation of HeART on the Backblaze dataset.

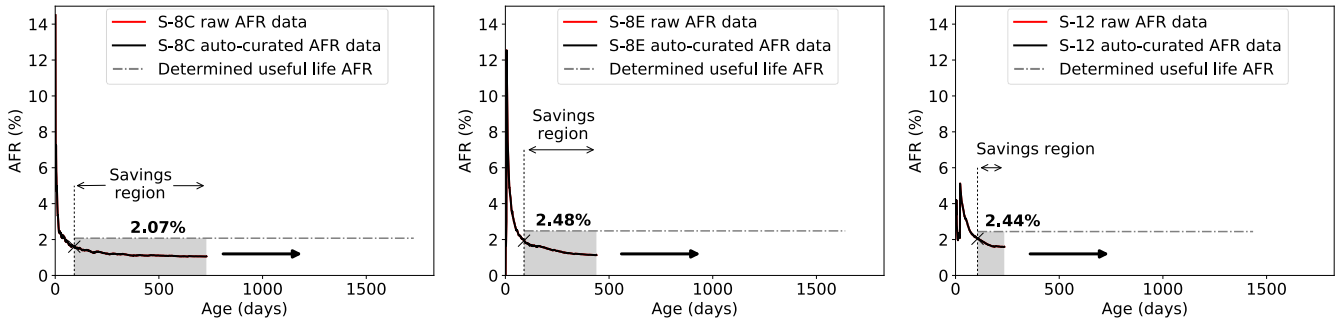
4.1 Implementation of the components

Our current implementation of HeART leverages existing, standard algorithms for anomaly detection and change point

[7] Although the H-4A graph in Fig. 5b appears to show a sudden, huge rise in *AFR*, we believe that it is an artifact of Backblaze's recording of decommissioned disks as failed, based on the device removal pattern seen in the failure data. Data from more sources are needed to confirm this hypothesis. If some disks do exhibit such transitions, then strategies for predicting failures (and wearout onset), such as by using S.M.A.R.T. statistics [4, 21, 44, 48], will be needed to use any but the most conservative redundancy schemes.



(a) S-4 with HeART-determined AFR **(b)** H-4A with HeART-determined AFR **(c)** H-4B with HeART-determined AFR



(d) S-8C with HeART-determined AFR **(e)** S-8E with HeART-determined AFR **(f)** S-12E with HeART-determined AFR

Figure 9: HeART in action on all disk groups, showing successful identification of infant mortality, useful life and wearout periods as well as automatic removal of anomalies.

detection. Employing more sophisticated algorithms might lead to even better results.

Anomaly detector: For anomaly detection, our current implementation of HeART uses the RRCF algorithm [3] exposed by Amazon's data analytics service offering called Kinesis [2].[8] The anomaly detector acts on a reliability data stream made available by the disk health monitoring system. The output from the anomaly detector is also a data stream containing anomaly scores produced by the RRCF algorithm. Potential anomalies identified by RRCF have a higher anomaly score than data that the algorithm considers non-anomalous. RRCF generates the anomaly score based on how different the new data is compared to the recent past. For consistency with change point detection, we set the window size of the recent past to be one month. If the anomaly score is above a certain threshold, HeART considers that snapshot of reliability data as anomalous. RRCF advises to only consider the highest anomaly scores as true anomalies [3]. The anomaly score threshold is a tunable parameter in HeART. Lowering the score makes HeART more sensitive to fluctuations in AFRs.

Change point detector: Our current implementation of HeART uses a standard window-based change point detection algorithm, which compares the discrepancy between adjacent *sliding windows* within the AFR curve to determine if a change point has been encountered. In particular, we employ the *Ruptures* library for online change point detection [39, 40]. We set the sliding window size to one month, because AFRs at a lower granularity than a month are jittery.

[8] We use Amazon's service so as to avoid re-implementing a state-of-the-art algorithm.

4.2 Evaluation on the Backblaze dataset

Identifying useful life period. Fig. 9 shows the results from HeART running on all 6 disk groups of the Backblaze dataset. HeART accurately identifies the infancy, useful life and wearout stages of the S-4, H-4A and H-4B disk groups shown in Figs. 9a, 9b and 9c, respectively. For the S-8C, S-8E and S-12E disk groups (Figs. 9d, 9e and 9f), HeART identifies the end of infancy and correctly shows that they are still in their useful life. The width of the shaded region of each disk group highlights the "savings region", i.e. the useful life period determined by HeART for which HeART potentially suggests a lower redundancy scheme. The height of the shaded region in Fig. 9 denotes the AFR values protected by the useful life AFR value determined by HeART for that disk group.

It is important to note that even though Fig. 9 shows cumulative AFR behavior, HeART performs anomaly detection and online change point detection on AFRs calculated using monthly sliding windows. Thus, not only is the cumulative AFR always inside the shaded region, but the instantaneous failure rate for any 30-day period is also less than the determined AFR value. In fact, the first rise in the instantaneous failure rate is what determines the end of the useful life period. Fig. 10 shows the instantaneous failure rate of S-4 disks being lower than the determined useful life AFR value throughout the useful life period.

Disk groups	r_{def} = $MTTDL^{(14,10)}_{4.01\%AFR} = 1.46E+21$				r_{def} = $MTTDL^{(9,6)}_{4.01\%AFR} = 3.31E+16$				r_{def} = $MTTDL^{(3,1)}_{4.01\%AFR} = 6.36E+12$			
DG	*AFR*	$MTTDL^{r_{def}}$	r_{DG}	$MTTDL^{r_{DG}}$	**Cost↓**	$MTTDL^{r_{def}}$	r_{DG}	$MTTDL^{r_{DG}}$	**Cost↓**	$MTTDL^{r_{def}}$	r_{DG}	$MTTDL^{r_{DG}}$	**Cost↓**
S-4	4.01%	$1.46E+21$	(14, 10)	$1.46E+21$	**NA**	$3.31E+16$	(9, 6)	$3.31E+16$	**NA**	$6.36E+12$	(3, 1)	$6.36E+12$	**NA**
H-4A	1.82%	$7.57E+22$	(24, 20)	$3.56E+21$	**14%**	$7.80E+17$	(15, 12)	$7.20E+16$	**16%**	$6.80E+13$	(4, 2)	$1.70E+13$	**33%**
H-4B	2.04%	$4.28E+22$	(24, 20)	$2.01E+21$	**14%**	$4.94E+17$	(15, 12)	$4.56E+16$	**16%**	$4.83E+13$	(4, 2)	$1.21E+13$	**33%**
S-8C	2.07%	$3.98E+22$	(24, 20)	$1.87E+21$	**14%**	$4.66E+17$	(15, 12)	$4.30E+16$	**16%**	$4.62E+13$	(4, 2)	$1.16E+13$	**33%**
S-8E	2.48%	$1.61E+22$	(21, 17)	$1.58E+21$	**11%**	$2.26E+17$	(13, 10)	$3.99E+16$	**13%**	$2.69E+13$	(4, 2)	$6.72E+12$	**33%**
S-12E	2.44%	$1.75E+22$	(21, 17)	$1.72E+21$	**11%**	$2.41E+17$	(13, 10)	$4.26E+16$	**13%**	$2.82E+13$	(4, 2)	$7.06E+12$	**33%**

Table 3: Disk space saved by HeART by tuning the redundancy in the useful life of a disk group according to the observed disk group-specific *AFR*s. The units for *MTTDL*s is years. The cost savings are calculated for 3 default schemes: (14, 10) on AFR 4.01% disks, (9, 6) on AFR 4.01% disks and 3-replication (i.e. (3, 1)) on AFR 4.01% disks. Thus, the target reliability is the *MTTDL* of the respective default redundancy schemes using a 4.01% AFR (the r_{def} table header). The max dimension of the scheme permitted during useful life for each disk group has at most twice the dimension of default redundancy scheme, i.e. 20 data chunks for $(14,10)$, 12 data chunks for $(9,6)$ and 2 data chunks for 3-replication.

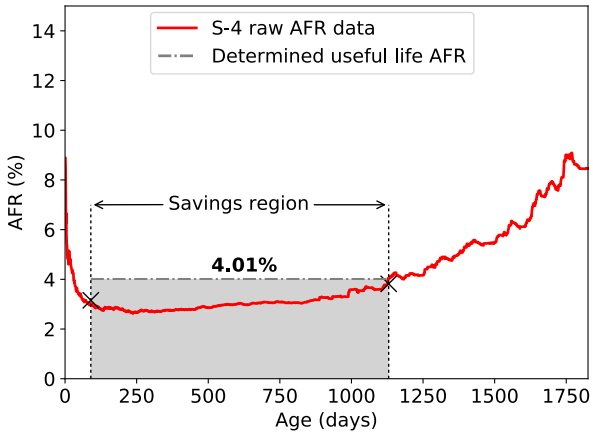


Figure 10: *AFR* of the S-4 disk group using a sliding window of 30 days. The determined useful life *AFR* value by HeART is conservative enough to subsume even the 30-day *AFR* values which vary more than the cumulative *AFR*s.

In contrast to S-4 (Fig. 9a), the H-4A (Fig. 9b) and H-4B (Fig. 9c) disk groups have a sudden occurrence of their respective wearout stages. The quick reactivity requirement explained in Section 3.1 comes into effect for these disk groups. How quickly HeART reacts to changes in the *AFR* is determined by how quickly failure data is provided to HeART. Since Backblaze maintains daily snapshots of disk health, the quickest reaction to an increased failure rate is on the day that the failures occur. In our evaluation, HeART successfully identifies the increased *AFR* on the very day it was provided with the increased *AFR* data.

Anomaly detection. As explained in Section 3.3, the anomaly detector successfully detects five anomalies in the lifetime of H-4B disks. Additionally, two anomalies are also detected for the H-4A disks. Correctly identifying anomalous events increased the identified useful life period of H-4B disks by over 5×. In the absence of anomaly detection, the end of useful life period would have been incorrectly identified at age 235 days (shown by point A in Fig. 7).

Cost savings per disk group. Table 3 summarizes the cost savings of employing disk group specific redundancy in their respective useful lifespans. Disk groups with similar *AFR*s are grouped together. As discussed in Section 2, we restrict the dimension (k) of the optimized code to at most 2× that of the default redundancy scheme (r_{def}). In each case of r_{def}, we set the target reliability to the *MTTDL* achieved by using the highest-*AFR* disk group, which in the case of Backblaze are the S-4 disks.

It is important to note that the useful life *AFR*s determined by HeART are higher than the useful life *AFR*s shown in Fig. 1. Recall from Section 2, that HeART adds a (tunable) buffer above the useful life *AFR* determined at the end of infancy (which is an additional 25% by default). HeART chooses to be conservative in determining a useful life *AFR* value to ensure that reliability targets are comfortably met and to elongate the length of the useful life period to maximize benefits.

As in Section 2, we exemplify the space reduction for erasure coding schemes using $(14,10)$ and $(9,6)$ schemes, which are known to have been employed in large-scale data centers [13, 25, 26, 28].

First, we evaluate using $(14,10)$ as the default redundancy scheme. $(14,10)$ has the lowest storage overhead (1.4×) among the default redundancy schemes we evaluate, making it the hardest to find codes that meet the target *MTTDL* and reduce overhead even further. Despite these constraints, HeART enables a 14% space reduction for H-4A, H-4B and S-8C disks by suggesting a $(24,20)$ code and a reduction of 11% for S-8E and S-12E disks by suggesting a $(21,17)$ code.

Next, we measure HeART's performance when using $(9,6)$ as the default redundancy scheme. We observe a space reduction of 16% on H-4A, H-4B and S-8C disks by using the maximum allowed $(15,12)$ redundancy scheme. For S-8E and S-12E disks, HeART suggests shorter $(13,10)$ code lengths compared to the above three disk groups in order to address their relatively higher determined *AFR* values, leading to a space reduction of 13%.

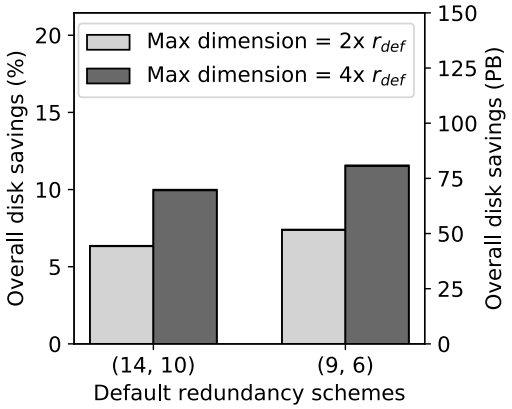


Figure 11: Overall space reduction achieved by HeART on the Backblaze dataset over the complete lifetime of every disk group, for erasure codes as the default scheme. For a maximum code dimension of up to $2 \times r_{def}$, we observe between $6-7.5\%$ space reduction and for a maximum code dimension of up to $4 \times r_{def}$, we observe between $10-12\%$ space reduction, translating to actual space savings of $40-80$ PBs.

Finally, we also include the cost reduction for the canonical redundancy scheme, 3-replication, for completeness. We see that HeART enables 33% space reduction for all disk groups. We note that if replication is employed primarily for availability, that data may not be a candidate for tuning redundancy through HeART.

For H-4A, H-4B and S-8C disks, HeART chose the $2\times$ max stripe-length for all three evaluated default redundancy schemes, extracting the maximum cost reduction (as explained in Table 2). Even with the maximum allowed stripe length, the $MTTDL$s for the above disks are approximately $2.5\times$ higher than the target $MTTDL$ value, suggesting further storage cost reductions if one is allowed even longer codes.

Overall cost reduction. To highlight the overall cost reduction achieved on the Backblaze disk fleet, we show the capacity-weighted cost savings in Fig. 11. This cost reduction is over the whole lifetime of the disks (including the unoptimized infancy and wearout periods) and for all six disk groups (including the unoptimized S-4 disks). We only show the benefits for the erasure coding schemes we evaluated, leaving out 3-way replication, since erasure codes are the more popular choice for data durability. The overall cost reduction achieved with the maximum stripe dimension being $2\times$ the default redundancy scheme is approximately 6% when using $(14,10)$ and approximately 7.5% when using $(9,6)$ as the default. If we relax the constraint of the maximum stripe dimension to $4\times$ the dimension of the default redundancy scheme, we can expect to achieve between $10-12\%$ overall space reduction. These modest percentage savings translate to significant savings in terms of actual storage space in large-scale clusters. For example, as shown on the right-side y-axis in Fig. 11, savings in storage space for the the Backblaze cluster range between $40-80$ PBs.

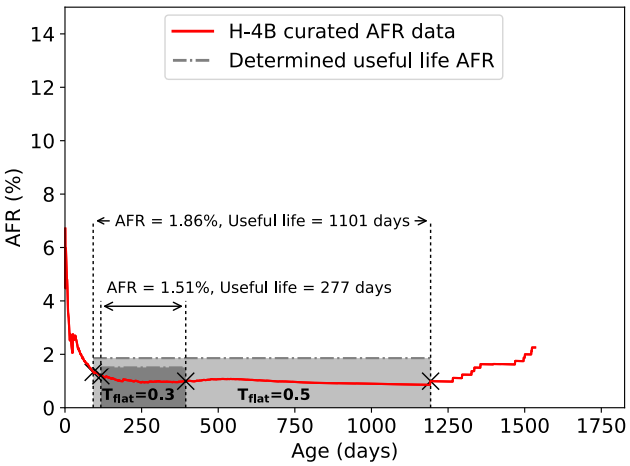


Figure 12: The effect of varying T_{flat} (AFR flatness threshold) on the H-4B disk group's AFR curve. Larger T_{flat} implies a higher useful life AFR along with a larger useful life period. The default value for T_{flat} in HeART is 0.5.

4.3 Sensitivity analysis

There are several configuration parameters that govern the behavior of HeART, of which most are dependent on the ready-made tools we have used for different components of our system (e.g., the threshold for anomaly scores when using RRCF for anomaly detection). There are, however, two fundamental parameters that are independent of which anomaly detector or change point detector is used.

Before going into the details about the two parameters, we note that the modulation of both the parameters only has an effect on the gains that our optimization can yield. *Neither of them affects correctness* of our framework or protection of data in any way. This allows operators of cluster storage systems to start with conservative values, observe the AFR behavior of their disks and accordingly choose apt values to minimize their costs without risking not reaching their reliability target. We next discuss the two parameters.

Flatness parameter (T_{flat}): T_{flat} is used to deduce the end of the infant mortality period. As mentioned in Section 3.4, the end of infancy is defined as the first 30+ day period beyond the change point detected after the first quarter such that the difference between maximum and minimum observed AFR is below the threshold T_{flat}. Thus, T_{flat} essentially determines the flatness in the AFR curve for a given period. Currently, we define T_{flat} to be 0.5. A larger T_{flat} value will reduce the length of infancy until it reaches 90 days, beyond which it will have no effect. A lower T_{flat} will enforce a stricter flatness criteria, typically causing end of infancy to be declared late. Ending infancy sooner potentially causes HeART to choose a larger value as the determined useful life AFR. This, in turn causes HeART to choose a stronger redundancy scheme (with higher space overhead) compared to one that would have been chosen with the determined AFR value derived as a result of a lower T_{flat} value. This reduces the achievable savings within the useful life period of the

disk group. As a tradeoff, we get a larger useful life period with a larger T_{flat}, since not only does infancy end sooner, but also the onset of wearout stage is postponed, since the increased useful life AFR now has higher tolerance to AFR variances throughout the useful life.

Fig. 12 shows the effect of varying T_{flat} on H-4B disks. We show the results for two different values $T_{flat} = 0.3$ and $T_{flat} = 0.5$. When T_{flat} was set to 0.3, we can see HeART declaring end of infancy at close to 100 days. Despite the buffer added to the determined useful life AFR at the end of infancy, the fluctuation in monthly AFRs caused a spike on day 394 to rise above the determined useful life AFR, causing HeART to announce end of useful life. In contrast, when $T_{flat} = 0.5$, infancy was declared to end on day 91, and the determined AFR value was high enough to tolerate the spike on day 394, increasing the useful life period by a significant amount.

Useful life AFR buffer: The AFR buffer is the conservative padding added to the useful life AFR determined at the end of infancy. Currently, the useful life AFR is determined as the AFR value at the end of infancy *plus an additional buffer*, the tunable AFR buffer parameter. The choice of the buffer value has similar tradeoffs to the flatness parameters discussed above. A high buffer value implies a more conservative approach to setting the determined useful life AFR. This will prolong the useful life period, but restrict the tuning of the redundancy scheme due to the high useful life AFR value determined (and thus reducing benefits). In contrast, setting a low buffer value will shorten the useful life period but allow more cost reductions during the useful life. Operators can set the buffer based on AFR fluctuations observed in their storage systems, which can stem anywhere from workload patterns to operational conditions.

5 Changes of the HeART (discussion)

HeART suggests redundancy schemes for use with each disk group during its useful life period, enabling safe redundancy tuning based on observed failure data. HeART recommends using the default redundancy scheme employed in the cluster during infancy and wearout periods. Exploiting HeART's recommendations in a cluster storage system requires some minor data placement policy changes and some online data redistribution. This section discusses both. Furthermore, since HeART changes redundancy schemes in response to the observed AFR curve, we also discuss estimating the required sample size (number of disks) for statistical confidence.

Data placement. HeART suggests per-disk-group redundancy schemes for hitting a particular data reliability target, based on observed AFRs. To use HeART safely, all data stored using a tailored redundancy scheme must be fully stored within the corresponding disk group—that is, all n chunks (data and parity chunks) of a stripe must be stored on disks within the same group. This restriction may be incompatible with some data placement schemes, such as Ceph's CRUSH [42, 43], and will add some complexity (conforming to disk group boundaries) to schemes that choose based on considerations like available capacity and load balancing.

Data redistribution. Many cluster storage systems include data redistribution mechanisms to deal with planned decommissioning and capacity/load balancing. Use of HeART will also require their use for transitioning from the default redundancy scheme (r_{def}) to a disk-group-specific scheme (r_{DG}), after infancy, and back again upon onset of wearout. Although this introduces extra redistribution load, we expect it to have a small impact on overall cluster load—at worst, it is two redistributions of the data over the 3–5 year deployment time of the disks.

Bulk changes should not be needed. On its face, HeART's redundancy scheme transitions appear to require massive redistributions, all at once. Not only would this be a load spike concern, not assuaged by the "not much load over the lifetime" argument, but it also potentially creates a capacity concern: a bulk transition from r_{DG} to the less space-efficient r_{def}, at the end of the useful life period, could require more space than is available. Fortunately, we do not expect this issue to arise in practice, as disks of a disk group are deployed over time rather than all at once (e.g., see Fig. 8) Since end of useful life is determined based on deployment age, rolling deployment will mean rolling wearout. Capacity exhaustion due to any given transition (of one subset of disks from one disk group to another) should not be as large a driver of slack capacity requirements, in practice, as other sources of variability (e.g., user demand). Furthermore, transitions from r_{def} to r_{DG} at the onset of useful life period can be gradually executed as this only reduces the achieved capacity savings and does not affect correctness or reliability guarantees.

Accurately characterizing bathtub curves. HeART is an online framework actively engaging with each disk group's bathtub curve. Naturally, it is important to understand how many disks one needs to observe before one can be confident of behavior of the bathtub curve of a particular disk group. There are several statistical bounds on the number of samples needed to reach a specified statistical confidence level. One technique is to use the *Chernoff-Hoeffding* Theorem [16, 1] to obtain a bound on the sample size (number of disks) required. For example, to achieve 99% confidence that the AFR of S-4 disks (which have an AFR of 3.29%, refer Table 2) is within the configured AFR-buffer of its determined useful life AFR (recall from Section 4, that the default AFR-buffer is an additional 25% over the useful life AFR value determined by HeART), the number of disks required is approximately 4,000. More advanced statistical techniques may provide tighter bounds and thereby indicate fewer required devices in a disk group.

6 HeART-less alternatives (related work)

The closest related work can be classified into disk reliability studies that identify reliability heterogeneity, techniques to predict disk failures using reliability data, and systems that automate redundancy scheme selection.

Numerous studies have been conducted to characterize disk failures [7, 10, 15, 18, 20, 23, 24, 29, 30, 31, 34]. Among the studies conducted on large production systems, Shah and Elerath [10, 34], Pinheiro et al. [24] and Ma et al. [20] independently verify that failure rates are highly correlated with disk manufacturers. These studies were conducted on the NetApp, Google and EMC disk fleets, respectively. Schroeder and Gibson also conducted a similar reliability study on disks from a high performance computing environment [30], not only highlighting reliability heterogeneity between disks deployed across systems, but also pointing out that disk datasheet reliability is very different from reliability observed in the field. Recently, Schroeder et al. [32] highlighted the heterogeneity in the reliability of different SSD technologies from four different manufacturers. Also, Schroeder et al. [29] reported heterogeneity of partial disk failures (sector errors) across makes/models for NetApp's disk fleet.

There have been numerous works that predict disk failures [14, 22, 36, 41, 47]. Among the more recent ones, Mahdisoltani et al. [21] use machine learning techniques to predict occurrence of partial disk errors using S.M.A.R.T. data. Anantharaman et al. [4] use random forests and recurrent neural networks to predict remaining useful life for HDDs. Both studies were performed on the Backblaze dataset.

Thereska et al. [37] built a self-prediction capability in cluster storage systems to assist in making informed redundancy and data placement decisions by answering *what-if* questions. It differs from HeART in that it does not perform and adapt to online analysis of reliability characteristics, relying on pre-knowledge of reliability metrics. Keeton et al. [19] built an optimization framework that automatically provided data dependability solutions to protect against site-level disasters by using information like workload patterns, and cost of recovery. This work also assumes prior knowledge of failure rates. Tempo [35] is a system that proactively creates replicas to ensure high durability in wide-area network distributed systems. It does this economically by allowing the user to specify a maximum maintenance bandwidth, and its design revolves around the efficient use of a distributed hash table. Carbonite [8] is a replica maintenance solution for distributed storage systems spread over the Internet, which makes efficient use of bandwidth in maintaining redundancy in the face of transient failures.

7 Conclusion

HeART enables more cost-effective data reliability for cluster storage systems. By robustly estimating per-disk-group *AFR*s and selecting the best redundancy settings for each, it avoids the space-inefficiency of one-size-fits-all redundancy schemes. Analysis of failure data for a large-scale production storage cluster shows that using HeART could achieve target data reliabilities with 11–33% fewer disks than popular configurations, offering huge potential cost savings.

Acknowledgements

We thank our shepherd Gala Yadgar and the anonymous reviewers for their valuable feedback and suggestions. We also thank the members and companies of the PDL Consortium (Alibaba, Amazon, Datrium, Dell EMC, Facebook, Google, Hewlett-Packard Enterprise, Hitachi, IBM, Intel, Micron, Microsoft Research, MongoDB, NetApp, Oracle, Salesforce, Samsung, Seagate, Two Sigma, Veritas, and Western Digital) and VMware for their interest, insights, feedback, and support.

References

[1] Chernoff-Hoeffding Theorem. `https://en.wikipedia.org/wiki/Chernoff_bound`.

[2] AMAZON. Kinesis. `https://aws.amazon.com/kinesis/`.

[3] AMAZON. Robust Random Cut Forest. `https://docs.aws.amazon.com/kinesisanalytics/latest/sqlref/sqlrf-random-cut-forest.html`.

[4] ANANTHARAMAN, P., QIAO, M., AND JADAV, D. Large Scale Predictive Analytics for Hard Disk Remaining Useful Life Estimation. In *2018 IEEE International Congress on Big Data (BigData Congress)* (2018).

[5] BACKBLAZE. Disk Reliability Dataset. `https://www.backblaze.com/b2/hard-drive-test-data.html`.

[6] BACKBLAZE. HDD SMART Stats. `https://www.backblaze.com/blog/what-smart-stats-indicate-hard-drive-failures`.

[7] BAIRAVASUNDARAM, L. N., GOODSON, G. R., PASUPATHY, S., AND SCHINDLER, J. An analysis of latent sector errors in disk drives.

[8] CHUN, B.-G., DABEK, F., HAEBERLEN, A., SIT, E., WEATHERSPOON, H., KAASHOEK, M. F., KUBIATOWICZ, J., AND MORRIS, R. Efficient Replica Maintenance for Distributed Storage Systems. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)* (2006).

[9] COLE, G. Estimating drive reliability in desktop computers and consumer electronics systems. *Seagate Technology Paper TP* (2000).

[10] ELERATH, J. Hard-disk drives: The good, the bad, and the ugly. *Communication of ACM* (2009).

[11] ELERATH, J. G. AFR: problems of definition, calculation and measurement in a commercial environment. In *IEEE Reliability and Maintenance Symposium (RAMS)* (2000).

[12] ELERATH, J. G. Specifying reliability in the disk drive industry: No more MTBF's. In *IEEE Reliability and Maintenance Symposium (RAMS)* (2000).

[13] FORD, D., LABELLE, F., POPOVICI, F. I., STOKELY, M., TRUONG, V.-A., BARROSO, L., GRIMES, C., AND QUINLAN, S. Availability in Globally Distributed Storage Systems. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (2010).

[14] HAMERLY, G., ELKAN, C., ET AL. Bayesian approaches to failure prediction for disk drives. In *International Conference on Machine Learning (ICML)* (2001).

[15] HEIEN, E., KONDO, D., GAINARU, A., LAPINE, D., KRAMER, B., AND CAPPELLO, F. Modeling and tolerating heterogeneous failures in large parallel systems. In *ACM / IEEE High Performance Computing Networking, Storage and Analysis (SC)* (2011).

[16] HOEFFDING, W. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association* (1963).

[17] HUANG, C., SIMITCI, H., XU, Y., OGUS, A., CALDER, B., GOPALAN, P., LI, J., YEKHANIN, S., ET AL. Erasure Coding in Windows Azure Storage. In *USENIX Annual Technical Conference (ATC)* (2012).

[18] JIANG, W., HU, C., ZHOU, Y., AND KANEVSKY, A. Are disks the dominant contributor for storage failures?: A comprehensive study of storage subsystem failure characteristics. *ACM Transactions on Storage (TOS)* (2008).

[19] KEETON, K., SANTOS, C. A., BEYER, D., CHASE, J. S., WILKES, J., ET AL. Designing for disasters. In *USENIX File and Storage Technologies (FAST)* (2004).

[20] MA, A., TRAYLOR, R., DOUGLIS, F., CHAMNESS, M., LU, G., SAWYER, D., CHANDRA, S., AND HSU, W. RAIDShield: characterizing, monitoring, and proactively protecting against disk failures. *ACM Transactions on Storage (TOS)* (2015).

[21] MAHDISOLTANI, F., STEFANOVICI, I., AND SCHROEDER, B. Proactive error prediction to improve storage system reliability. In *USENIX Annual Technical Conference (ATC)* (2017).

[22] MURRAY, J. F., HUGHES, G. F., AND KREUTZ-DELGADO, K. Hard drive failure prediction using non-parametric statistical methods. In *Springer Artificial Neural Networks and Neural Information Processing (ICANN/CONIP* (2003).

[23] PATTERSON, D. A., GIBSON, G., AND KATZ, R. H. A case for redundant arrays of inexpensive disks (RAID). In *ACM International Conference on Management of Data (SIGMOD)* (1988).

[24] PINHEIRO, E., WEBER, W.-D., AND BARROSO, L. A. Failure Trends in a Large Disk Drive Population. In *USENIX File and Storage Technologies (FAST)* (2007).

[25] RASHMI, K. V., SHAH, N. B., GU, D., KUANG, H., BORTHAKUR, D., AND RAMCHANDRAN, K. A Solution to the Network Challenges of Data Recovery in Erasure-coded Distributed Storage Systems: A Study on the Facebook Warehouse Cluster. In *USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage)* (2013).

[26] RASHMI, K. V., SHAH, N. B., GU, D., KUANG, H., BORTHAKUR, D., AND RAMCHANDRAN, K. A hitchhiker's guide to fast and efficient data reconstruction in erasure-coded data centers. *ACM Special Interest Group on Data Communication (SIGCOMM)* (2014).

[27] SAITO, Y., FRØLUND, S., VEITCH, A., MERCHANT, A., AND SPENCE, S. FAB: building distributed enterprise disk arrays from commodity components. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2004).

[28] SATHIAMOORTHY, M., ASTERIS, M., PAPAILIOPOULOS, D., DIMAKIS, A. G., VADALI, R., CHEN, S., AND BORTHAKUR, D. Xoring elephants: Novel erasure codes for big data. In *International Conference on Very Large Data Bases* (2013).

[29] SCHROEDER, B., DAMOURAS, S., AND GILL, P. Understanding latent sector errors and how to protect against them. *ACM Transactions on Storage (TOS)* (2010).

[30] SCHROEDER, B., AND GIBSON, G. A. Disk failures in the real world: What does an mttf of 1, 000, 000 hours mean to you? In *USENIX File and Storage Technologies (FAST)* (2007).

[31] SCHROEDER, B., AND GIBSON, G. A. Understanding failures in petascale computers. In *Journal of Physics: Conference Series* (2007), IOP Publishing.

[32] SCHROEDER, B., LAGISETTY, R., AND MERCHANT, A. Flash Reliability in Production: The Expected and the Unexpected. In *USENIX File and Storage Technologies (FAST)* (2016).

[33] SEAGATE. Hard disk drive reliability and MTBF / AFR. `http://knowledge.seagate.com/articles/en_US/FAQ/174791en`.

[34] SHAH, S., AND ELERATH, J. G. Disk drive vintage and its effect on reliability. In *IEEE Reliability and Maintenance Symposium (RAMS)* (2004).

[35] SIT, E., HAEBERLEN, A., DABEK, F., CHUN, B.-G., WEATHERSPOON, H., MORRIS, R. T., KAASHOEK, M. F., AND KUBIATOWICZ, J. Proactive Replication for Data Durability. In *USENIX International Workshop on Peer-to-Peer Systems (IPTPS)* (2006).

[36] STROM, B. D., LEE, S., TYNDALL, G. W., AND KHURSHUDOV, A. Hard disk drive reliability modeling and failure prediction. *IEEE Transactions on Magnetics* (2007).

[37] THERESKA, E., ABD-EL-MALEK, M., WYLIE, J. J., NARAYANAN, D., AND GANGER, G. R. Informed data distribution selection in a self-predicting storage system. In *IEEE International Conference on Autonomic Computing (ICAC)* (2006).

[38] TRIVEDI, K. *Probability and Statistics with Reliability, Queueing, and Computer Science Applications*. Wiley, 2001.

[39] TRUONG, C., OUDRE, L., AND VAYATIS, N. A review of change point detection methods. In *arXiv:1801.00718v1 [cs.CE]* (2018).

[40] TRUONG, C., OUDRE, L., AND VAYATIS, N. ruptures: change point detection in python. In *arXiv:1801.00826v1 [cs.CE]* (2018).

[41] WANG, Y., MA, E. W., CHOW, T. W., AND TSUI, K.-L. A two-step parametric method for failure prediction in hard disk drives. *IEEE Transactions on industrial informatics* (2014).

[42] WEIL, S. A., BRANDT, S. A., MILLER, E. L., LONG, D. D., AND MALTZAHN, C. Ceph: A scalable, high-performance distributed file system. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (2006).

[43] WEIL, S. A., BRANDT, S. A., MILLER, E. L., AND MALTZAHN, C. CRUSH: Controlled, scalable, decentralized placement of replicated data. In *ACM / IEEE High Performance Computing Networking, Storage and Analysis (SC)* (2006).

[44] XU, Y., SUI, K., YAO, R., ZHANG, H., LIN, Q., DANG, Y., LI, P., JIANG, K., ZHANG, W., LOU, J.-G., ET AL. Improving service availability of cloud systems by predicting disk error. In *USENIX Annual Technical Conference (ATC)* (2018).

[45] YANG, J., AND SUN, F.-B. A comprehensive review of hard-disk drive reliability. In *IEEE Reliability and Maintenance Symposium (RAMS)* (1999).

[46] ZHANG, Z., WANG, A., ZHENG, K., MAHESWARA, G. U., AND VINAYAKUMAR, B. Introduction to hdfs erasure coding in apache hadoop. *blog.cloudera.com* (2015).

[47] ZHAO, Y., LIU, X., GAN, S., AND ZHENG, W. Predicting disk failures with HMM-and HSMM-based approaches. In *Springer Industrial Conference on Data Mining (ICDM)* (2010).

[48] ZHU, B., WANG, G., LIU, X., HU, D., LIN, S., AND MA, J. Proactive drive failure prediction for large scale storage systems. In *IEEE/NASA Goddard Conference on Mass Storage Systems and Technologies (MSST)* (2013).