Disk Infant Mortality in Large Storage Systems

Qin Xin[1] qxin@cs.ucsc.edu
Thomas J. E. Schwarz, S. J.[1,2] tjschwarz@scu.edu
Ethan L. Miller[1] elm@cs.ucsc.edu

[1]*Storage Systems Research Center, University of California, Santa Cruz, CA 95064*
[2]*Computer Engineering Department, Santa Clara University, Santa Clara, CA 95053*

Abstract

As disk drives have dropped in price relative to tape, the desire for the convenience and speed of online access to large data repositories has led to the deployment of petabyte-scale disk farms with thousands of disks. Unfortunately, the very large size of these repositories renders them vulnerable to previously rare failure modes such as multiple, unrelated disk failures leading to data loss. While some business models, such as free email servers, may be able to tolerate some occurrence of data loss, others, including premium online services and storage of simulation results at a national laboratory, cannot.

This paper describes the effect of infant mortality on long-term failure rates of systems that must preserve their data for decades. Our failure models incorporate the well-known "bathtub curve," which reflects the higher failure rates of new disk drives, a lower, constant failure rate during the remainder of the design life span, and increased failure rates as components wear out. Large systems are vulnerable to the "cohort effect" that occurs when many disks are simultaneously replaced by new disks. Our more accurate disk models and simulations have yielded predictions of system lifetimes that are more pessimistic than existing models that assume a constant disk failure rate. Thus, larger system scale requires designers to take disk infant mortality into account.

1. Introduction

Data sets containing hundreds or thousands of terabytes are commonly used today in diverse applications such as simulation data at a national lab, satellite data at a national agency, and a web-based email provider. Advances in disk drive capacity and the decrease in the cost per gigabyte for disk storage have combined to enable the use of a disk farm rather than a tape library to store such large data sets. A smaller installation might consist of 3000 one terabyte disk drives with a total user capacity of 1–2 petabytes, exclusive of redundancy information, and might grow at the rate of several hundred terabytes each year. Some applications, such as free email, can afford to be cavalier about occasionally losing a small amount of data, but often, valuable data can only be reproduced at high costs and even small losses can make large amounts of data unusable. Thus, most large-scale storage systems need to be very reliable.

New phenomena emerge with the sheer size of storage repositories. For example, protection by simply using RAID 1 or RAID 5 is frequently insufficient because the chance that *some* RAID array in the system will experience data loss is too high [30]. In large systems, variability in the failure rate of the disks in a system can contribute to high failure rates because of the possibility that a single piece of data may, by chance, be stored on several devices with a relatively high failure rate. It is well-known that disk drives fail at higher rates early in their lives, with the failure rate dropping during the first year and remaining relatively constant for the remainder of the disk's useful lifespan, rising again at the end of the disks lifetime. Because of its shape, the failure rate curve is called a "bathtub" curve [12, 23]. A typical disk is obsolete before its failure probability starts to climb, in part because newer drives have much higher capacity and performance; thus, the tail of the bathtub curve is of little practical importance. As we will see, this is not true at the beginning of the curve, resulting in a phenomenon known as *infant mortality*. The International Disk Drive Equipment and Materials Association (IDEMA) recently proposed a more sophisticated way to measure disk drive reliability by using four different MTBF values for disks aged 0–3 months, 3–6 months, 6–12 months, and one year to the end of design life span [9, 27].

In the remainder of this paper, we show that disk infant mortality becomes an important factor in very large storage systems. We argue that autonomous data management must be based on disk age as well as data

This paper was published in the *13th Annual IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2005)*, Atlanta, GA, September, 2005.

characteristics, such as those used by AutoRAID [29]. This is particularly true for large-scale storage systems such as the Federated Array of Bricks [10], IceCube [17], FARSITE [1], the Google File System [11] and OceanStore [19], all of which require high system reliability with no data loss over a long system lifespan.

2. Related Work

Reliability is one of the key characteristics of a storage system. However, in the words of Anderson, *et al.* [2], reliability is "one of the trickiest drive characteristics to measure" because of complex factors including duty hours, temperature, altitude, and spindle starts and stops [9]. Not all disks are equally reliable—commercial grade SCSI disks tend to be much more reliable than consumer grade IDE and ATA disks [26].

An accurate disk reliability model does not yet exist because the exact distribution of disk drive failure rates is not known. Even though the bathtub curve is noted in several early RAID papers [4, 12, 23], many researchers assume a constant disk failure rate [3, 6, 13, 16, 22, 24]. Disk drive reliability is often quoted as a single value, the Mean Time Between Failure (MTBF). Recently, Elerath and IDEMA [9, 27] proposed a more detailed MTBF rating, consisting of four different values corresponding to drive ages of 0–3 months, 3–6 months, 6–12 months, and one year to End of Design Life (EODL).

Prior research into storage system reliability falls into one of two categories: modeling of relatively small-scale storage systems, and high-level analysis of large-scale storage systems. RAID system reliability has long been modeled using Markov models [4, 7, 3, 13, 16, 22], and Carrasco recently used a bounded technique to solve an involved model for the availability of a RAID-5 system [5]. Dugan and Ciardo [8] modeled a replicated file system using Petri nets, which were in turn solved by a Markov model generated from the Petri net. Markov modeling has also been used to provide dependability estimates for cache-based RAID controllers [18] and video on demand systems [20]. Weatherspoon and Kubiatowicz [28], on the other hand, studied the effects of using replication and erasure coding on the long-term reliability of Oceanstore [19], a global-scale storage system in which data was very highly replicated and time to failure detection was on the order of months. Data in such a system survives by keeping sufficient redundancy information to survive the loss of dozens of storage servers, as was done in Glacier [14]. Since Markov models of a multi-exabyte storage system that can survive the failure of over twenty disks simultaneously are likely to be unsolvable, both the Oceanstore and Glacier researchers used simple probability theory to estimate system reliability. None of these reliability studies took infant mortality into account; fortunately, the inclusion of infant mortality in reliability models has only a small effect on the small-scale storage systems that were the focus of most studies, though its influence on larger systems is more significant, as we will show.

3. System Architecture

While our results and analysis are valid for both traditional block devices and object-based storage devices (OSDs), we describe them in terms of an OSD-based storage system. In such a system, data and metadata are lumped together as objects stored on OSDs. On each OSD, a number of objects are collected into groups of approximately equal size. The groups are then stored redundantly in the system. We call a group of data blocks composed of user data and their associated replicas or parity / erasure code blocks a *redundancy group*. To achieve this redundancy we can employ various strategies that differ in timing, storage use, and availability. A system is *k*-available if the system can tolerate *k* disk failures while preserving access to all its data. For instance, to achieve 1-availability, we can simply mirror a group on another disk in the system. We can also assemble *m* redundancy groups in a reliability stripe and append a parity group that contains the bitwise parity of all the other groups to the stripe, similar to the technique used in RAID 5. The two schemes differ in the complexity of updates and recovering data after a single failure. The need for more reads to recover data in the RAID 5 scheme leads to longer recovery times and hence to a greater vulnerability to further disk failures that create data loss. To achieve higher levels of availabilities, we can use several different schemes, including more replicas, additional parity groups for a reliability stripe using techniques such as Reed-Solomon coding, and assignment of an object group to more than one reliability stripe.

At any given level of availability, the speed of repair after a failure determines reliability for a specific storage system. Modern disk drives have hundreds of gigabytes of capacity and it takes a long time to rebuild data on a failed disk. For example, rebuilding a 500 GB disk takes as long as 17 hours. To make matters worse, the increase in disk capacity has outpaced the increase in disk bandwidth, making disk repair time even longer as storage densities increase. In order to achieve high-speed data repair, we proposed FAst Recovery Mechanisms (FARM) [31], a technique that declusters redundancy groups across multiple disk drives in order to speed up recovery from disk failure. By using declustering, replicas and parities for the objects stored on any individual OSD are distributed across a set of redundancy groups so that the objects on a failed disk drive can be reconstructed by reading the remaining pieces (replicas or data and parity) stored on multiple disks (the "data recovery sources") and then writing the repaired objects to multiple disks (the "data recovery targets"). By utilizing the bandwidth of multiple data recovery sources and targets, the process of data recovery can be done in a distributed fashion, thus shrinking the windows of vulnerability of data losses.

In the remainder of this paper, we refer to a system that uses distributed recovery as a system with FAst Recovery Mechanism (FARM). Comparatively, a non-FARM system is a traditional system that reconstructs the failed data objects on a single disk drive. In principle, a system with FARM has shorter disk repair time than a system without FARM.

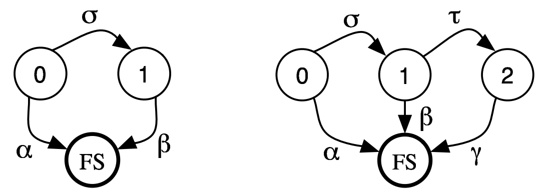
4. Modeling Infant Mortality

We propose hidden Markov models that consider high failure rates at the early stage of disk lifetime and use them in modeling failures of a single disk drive. Based on our failure models of a single drive, we further propose a system-level Markov model, and use it to observe the cohort effect brought by the high failure rates of a number of young disks.

4.1. Single Disk Modeling

A disk is a complicated device, consisting of many electronic, magnetic, mechanical, and chemical components. The manufacturing process is complicated and some errors show up only some time into the lifespan of the product. For this reason, disk manufacturers *burn in* new disks. Since a manufacturer needs to format their disks at a low level and mark faulty blocks, a disk has already been operated for some hours before it is shipped to the end user. Burn-in periods for the more expensive and higher-performing SCSI drives are longer, but much more limited for the commodity IDE drives; however, even some SCSI disks are "dead on arrival." A true disk failure model would faithfully reflect the possible states of each disk component after production and calculate the lifespan of a disk by finding the first combination of conditions that lead to death. The complexity of the disk renders this approach impossible, (but see Shah and Elerath [25]). In lieu of modeling disk failure by modeling their causes, we present a simple Markov model for disk failure that predicts more reasonable failure rate behavior.

4.1.1. Measurements Data on failures are hard to come by and they tend to be poor. Disk producers have to rely on data provided by customer feedback for field data, and are reluctant to release failure rates for fear of the negative impressions such information might give about their products. To predict failure rates in a normal use environment, drive manufacturers use life-acceleration, during which they subject disks to a rigorous "exercise regimen." Additionally, they use slightly different internal definitions of failure events and for that reason alone are reluctant to make data public.



(a) Three-state model. (b) Four-state model.

Figure 1. Hidden Markov models for disk failure.

In general, the disk drive industry only advertises MTBF (Mean Time Between Failures). However, users tend to observe higher failure rates than expected. Sometimes the conditions under which disks are deployed are worse than the disk manufacturers assume when giving the MTBF values, but the fact that failure rates vary over the lifespan of a disk plays another important role. For this reason, the reliability standard R2-98 [27] gives disk failure rates for four different periods, namely 0–3 months, 3–6 months, 6–12 months, and one year to the End of Design Life (EODL).

4.1.2. Realistic Disk Failure Models Real disk failure rates are smooth and not a splice of constant functions—there is no sudden jump in failure rate after each three month period. In order to determine the influence of the exact shape of the disk failure rate function, we generate several "realistic functions." First, we use a simple Hidden Markov Model to generate smooth failure rates. The simplicity of this model with its few states allows us to model large systems. In the three state model in Figure 1(a), State 0 describes a brand new disk, State 1 a burnt-in disk, and State FS is the failure state. The failure probability is higher in State 0 than in State 1. As a result, the transition rate α from State 0 to State FS is higher than the transition rate β from State 1 to State FS. The model in Figure 1(b) introduces an intermediate, "burning in" state. For both models, we generated parameters that closely fit the IDEMA values shown in Table 2 by calculating a formula for the IDEMA values from our parameters and then using iterative improvement to solve the resulting non-linear equations. We insisted, however, that the failure rate from 0 to 72 months was exactly that of the example values. The parameter σ in the three state model determines the burn-in period. At time $1/\sigma$,

parameters / model type	α (%/1000h)	β (%/1000h)	γ (%/1000h)	σ (/year)	τ (/year)
HMM-3state	0.618059	0.198044	-	2.796275	-
HMM-4state	0.350385	0.636888	0.198788	53.450	3.0400

Table 1. Parameters of three- and four-state Markov model.

time period (month) / model type	0–3	3–6	6–12	12–72	0–72
HMM-3state	0.5	0.347508	0.253143	0.199810	0.222917
HMM-4state	0.500108	0.349901	0.250027	0.1999988	0.222917
IDEMA	0.5	0.35	0.25	0.2	0.222917

Table 2. Model fits of three- and four-state Markov model, failure rate measured in %/1000 hours.

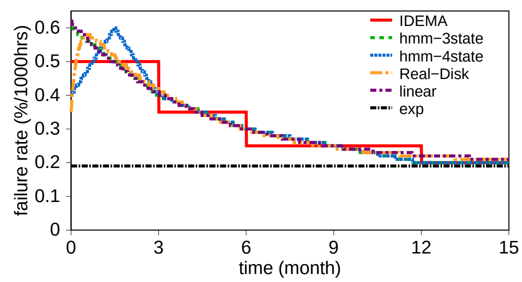
about 63% of all disks are burned in and the failure rate is $0.37\alpha + 0.63\beta$. Since disk failure rates seem to become constant after about 12 months, σ should have a value larger than 1. The parameter β is the terminal failure rate (recall that we do not model disks that are used long enough to suffer an increase in failure rate towards the end) and should be slightly smaller than the 1-year-to-EODL value. As should be expected, three parameters (α, β, σ) cannot reproduce the four free IDEMA failure values that determine the overall failure rate, but they can come quite close. The additional two parameters of the four state model allow a much closer reproduction.

Not all failure rates of "actually existing" disks show the same development of failure rates as they age. In some cases, the disk failure rate starts out low, but increases sharply for a short amount of time, and then falls from its peak very much like our Hidden Markov Model failure rates. We used linear splines in order to generate the failure rate for this "Real Disk" shown in Figure 2(a). In addition, we created a linearly decreasing disk failure rate in the first year of disk lifespan, called "Linear" in Figure 2(a). We show our simulation results of data loss probability under the varied disk failure models in Section 5.2.

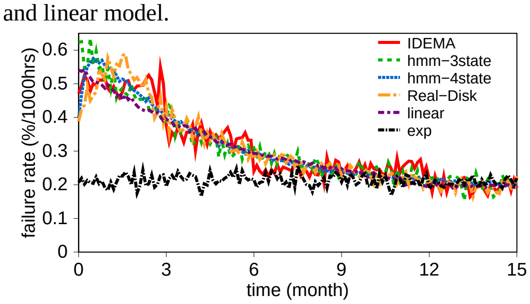
Our results taken together indicate that the exact shape of the failure rate is less important than the mere existence of infant mortality. This is thus a vindication of the IDEMA approach that captures a continuous failure rate in four discrete values.

4.2. System-level Markov Models

Based on the model of a single disk, we can also use a hidden Markov model to model a large system. Since the three-state model (new, burnt-in, failed) of a disk seems to provide a reasonably good model for thousands of disks, we use it as the base for system modeling. Our approach is to model a system with N living disks as $N+1$ states (i, j), where i denotes the number of new disks, j the number of burnt-in disks, and $N = i + j$ is the total number of disks; this model would require $N(N+3)/2$ different states to model up to N failures. Old (and thus burnt-in) disks are replaced at rate δ; thus, $1/\delta$ is the economic lifespan of a disk. To limit the number of states, we make the unrealistic assumption that upon failure and reconstruction of the contents of the disk, another disk (either new or burnt



(a) Failure rate by IDEMA model, realistic disk model and linear model.



(b) Simulated disk failure rate in the first 15 months under various disk failure models, based on 20,000 samples.

Figure 2. Comparison of failure rate model of a single disk: the stair-step model proposed by IDEMA, three- and four-state hidden Markov models, real disk model, linear model, and exponential model.

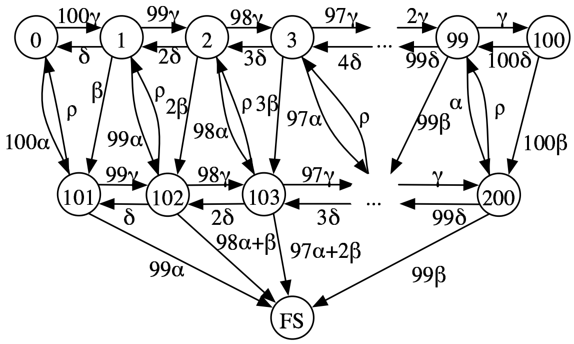


Figure 3. Markov model for a system with 100 disk drives.

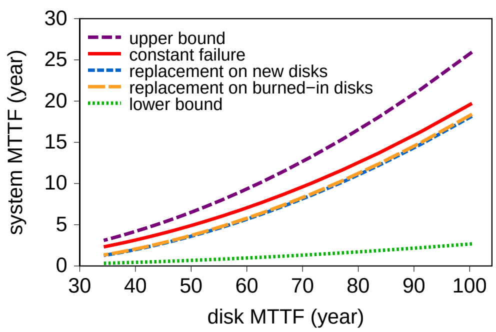


Figure 4. System MTTF for a 3000 disk system using FARM. The system is single failure resilient, but will lose data if a second drive fails before the data from the first one is recovered.

in) is added to the system. If, in addition, we only model systems that survive a limited number of disk failures within the window of vulnerability constituted by previous failures, the resulting Markov model is both reasonable accurate and solvable for relatively large storage systems. For example, Figure 3 shows the Markov model for a one-available system that uses FARM.

Figure 4 shows the cohort effect on a system with 3,000 drives. We compare two different disk failure rates, a variable failure rate, incorporating infant mortality, and a constant failure rate that ignores infant mortality, both with the same disk MTBF. In our system, failed disks can be replaced by either a new disk or a burned-in disk; the resulting system MTTF is nearly identical for the two cases. Figure 4 also includes lines corresponding to disks that fail either with the initial failure rate α (the lower bound for MTTF) or the terminal failure rate β (the upper bound for MTTF). It is important to note that ignoring infant mortality will result in overestimating total system MTTF, as shown by the divergence between the "constant failure" line and the lines that show the system MTTF using our more realistic model. The divergence between the upper and lower bounds and the more realistic models is more pronounced as disk lifetime increases. We calculated MTTF values by inverting very large matrices with LU decomposition so that we could derive a bound for the numerical error which turned out to be negligible. One of the more sophisticated methods for solving large Markov systems needs to be used for systems with tens of thousands of disks or for systems that survive more failures.

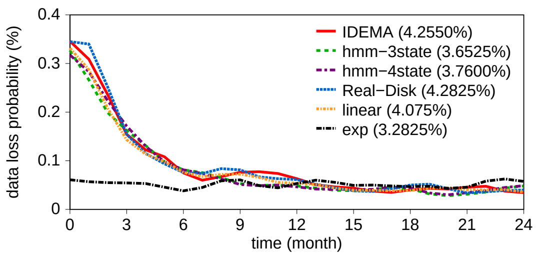
5. Effects and Mitigation of Infant Mortality

Using simulations, we explored the effects of infant mortality in large-scale storage systems. We first varied the number of disk drives in a storage system and found that the distribution of data loss probability over six years differs when infant mortality is taken into account. Next, we studied the various disk replacement strategies, and showed which strategies provided the best overall system reliability when infant mortality was considered. Based on these studies, we propose an adaptive data redundancy scheme to reduce the impact of infant mortality.

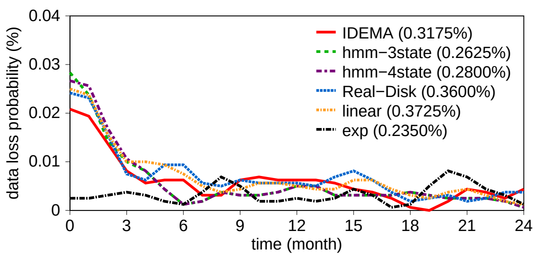
5.1. Simulation Methodology

To examine the effect of infant mortality on large storage systems, we ran discrete event-driven simulations built with the PARSEC simulation tool [21].

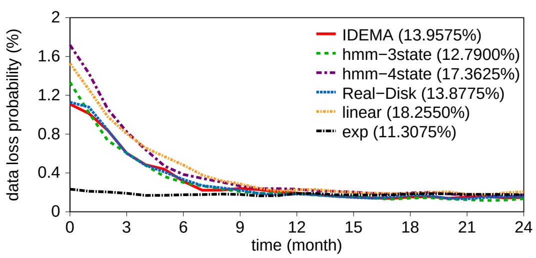
In a simulated storage system, data is distributed randomly by the RUSH [15] placement algorithm, which probabilistically guarantees that data will be distributed evenly across all of the disk drives. RUSH supports data redundancy and gives a list of disk IDs where the current pieces of each redundancy group reside on along with the additional disk IDs which will be used as the locations of the recovery targets during data recovery processes. We injected the varied failure models for individual disk drives, including IDEMA, hidden Markov models, and the exponential model, into the simulated storage system composed of thousands of disks. Using these models, the time to disk failure can be simulated; this time is then used to schedule the failure events. Whenever a failure event occurs, a data recovery event is triggered and the data on the failed disk is copied from the recovery sources to the recovery targets. When one or more additional failures happen during the data recovery process and the data cannot be rebuilt anymore, we count it as one occurrence of data loss and record the timestamp.



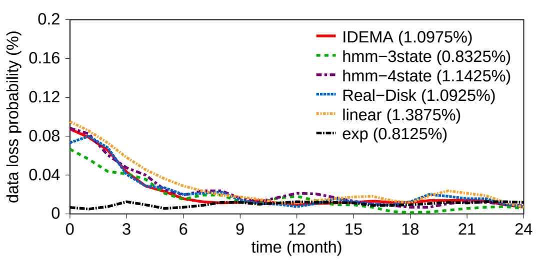
(a) Non-FARM, two-way mirroring.



(b) FARM, two-way mirroring.

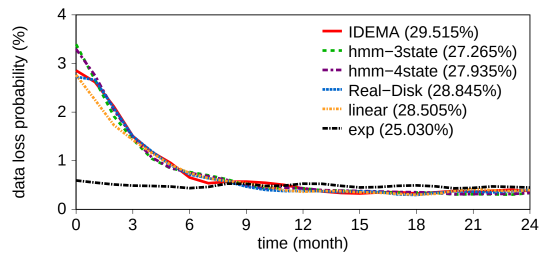


(c) Non-FARM, RAID 5

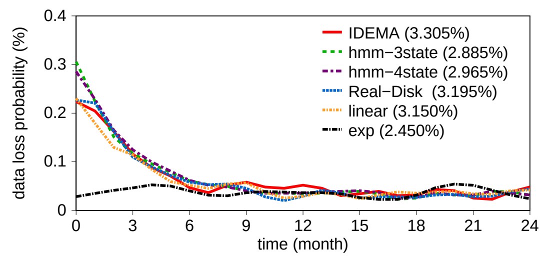


(d) FARM, RAID 5

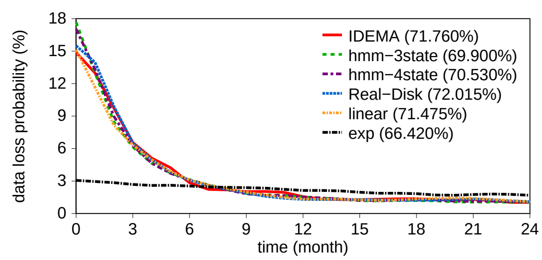
Figure 5. Distribution of data loss probability for a system with 1000 disk drives over six simulated years. (Note: Y-axis scale is different.)



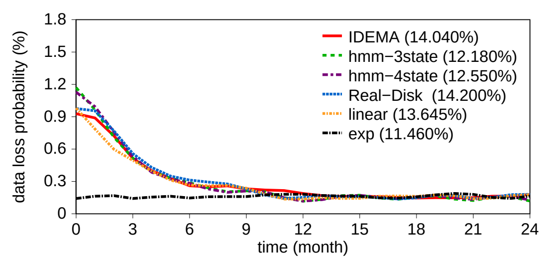
(a) Non-FARM, two-way mirroring.



(b) FARM, two-way mirroring.



(c) Non-FARM, RAID 5



(d) FARM, RAID 5

Figure 6. Distribution of data loss probability for a system with 10,000 disk drives over six simulated years. (Note: Y-axis scale is different.)

We varied several parameters in our simulation, including the number of disk drives in a simulated system, the disk failure model, data redundancy configuration (mirroring or RAID 5-like parity scheme), and data recovery method (distributed recovery with FARM or traditional disk rebuilding method that we term "Non-FARM"). For each configuration, we simulated the system over a period of six years and repeated the simulation for thousands of times to gather data samples. The data loss probability was calculated as the total occurrence of data loss divided by the total number of simulated samples. By using the recorded timestamps of data loss occurrence, we also calculated the distribution of times when data was lost and plot the probability density function curves of the data loss distribution.

5.2. Effects of Infant Mortality

We simulated systems with various redundancy mechanisms and either 1,000 disks or 10,000 disks over

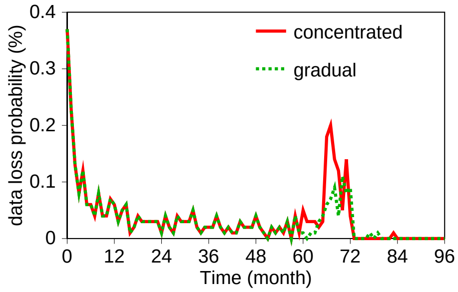


Figure 7. Distribution of System Data Loss under Two Disk Replacement Strategies.

a period of six years, assuming that initially all drives were not burnt in, with the results shown in Figures 5 and 6. Two kinds of data redundancy schemes were simulated: two-way mirroring and 4+1 RAID 5. We configured the system with FARM, described in Section 3) and without FARM respectively. The figures only include the first 24 months in the graphs to emphasize the differences between the naïve model of disk failure and our model that includes infant mortality. After the first 24 months, the data loss probability remains stable since disk drives become mature and their failure rates stay the same after the infant period until the end of disk lifespan (six simulated years). In the graphs, the percentages on the labels give the data loss probability over six years for each model. We observed the presence of higher data loss probability during the first 12 months under the disk failure models that take infant mortality into consideration. This effect does not appear under the exponential model which assumes disk failure rate remains unchanged in a disk's lifespan. The effect of infant mortality is more pronounced for larger systems, as can be seen by comparing the systems with 1,000 and 10,000 disks, implying that designers of large storage systems that store petabytes of data must consider the effects of infant mortality. Note that the Y-axis scales in Figure 5 and 6 are not the same. We also observed that our system reliability did not differ considerably under the five models of infant mortality: IDEMA, 3-state hidden Markov, 4-state hidden Markov, "real disk," and linear model. This shows that, although disk infant mortality itself is very important, the precise shape of the failure rate curve is less crucial.

Our generic system distributes the objects in a reliability group over all the disks in the system. Under these conditions, if a disk dies, its contents are reconstructed by reading from all the surviving disks in the system and written to the same set of disks. For many reasons, including modularity and easy administration, we might want to localize the placement of reliability groups; thus, we explored the effects of infant mortality on such a restriction. To do this, we constrained the distribution of objects in a reliability group to a *cluster* of disks; the resulting simulation modeled a storage system with $N = 10000$ disks, divided into C clusters with N/C disks each. Within each cluster, we used mirroring (RAID 1) and RAID 5. As expected, our simulation suggested that clustering had no significant influence on the system failure rate.

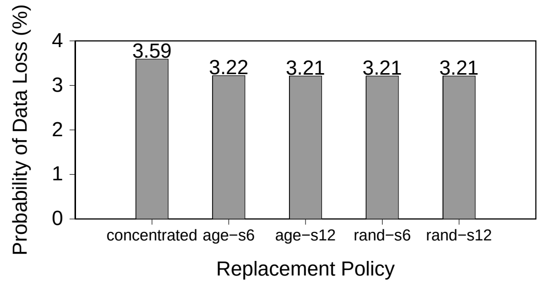


Figure 8. Disk Replacement Strategies.

5.3. Disk Replacement Strategy

Disk replacement strategy becomes an important issue for a large-scale storage system due to the cohort effect caused by infant mortality. For example, a new storage system with 10,000 disk drives and two petabytes of data capacity will lose about 10% of the disks due to premature (unplanned) disk failure during the disks' design life span. The disks to replace these failed disks will be added in batches. The remaining 9,000 disk drives must be replaced when they reach their EODL, resulting, again, in a system with many young disk drives. While the failure rate of individual drives does not increase much, system failure time does increase dramatically as the average age of disks in the system is reset. The system failure rate jumps when we concentrate disk drive replacement—as Figure 7 shows, the peak likelihood of data loss occurs during the drive replacement period. A more gradual policy spreads the replacement out and avoids the marked jump. We experimented with two gradual replacement policies: *age-based* and *random* replacement. In the age-based policy, disks to be replaced in a given year are identified at the start of the year; 1/12 of this pool is replaced each month, replacing the oldest disks first. To measure sensitivity to granularity, we also investigate a bimensal policy. In the random policy, we replaced disk drives completely randomly regardless of their age at a rate that on average lets disks stay the length of the designed life cycle. Again, we replace in batches; either every month or every other month.

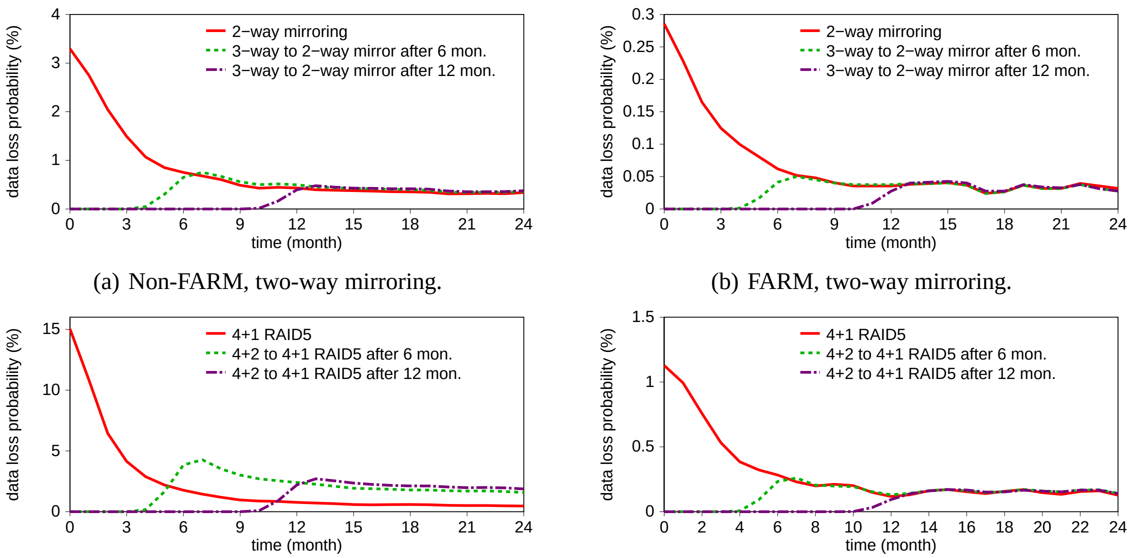
Table 3 lists the data loss probability over ten years under various disk replacement policies. When we re-

Replacement Policy	Concentrated	Age-based		Random	
		split-6	split-12	split-6	split-12
Data Loss Probability	3.59%	3.22%	3.21%	3.22%	3.21%

Table 3. Data Loss Probability under Various Disk Replacement Strategies.

Table 4. Data Loss Probability for Adaptive Redundancy Schemes.

	2×mirror	3 → 2× after 6 mo.	3 → 2× after 12 mo	4+1 RAID5	4+2 → 4+1 after 6 mo.	4+2 → 4+1 after 12 mo.
Non-FARM	27.935%	21.02%	18.38%	70.53%	62.14%	58.17%
FARM	2.96%	1.96%	1.72%	12.55%	9.12%	7.96%



(a) Non-FARM, two-way mirroring.

(b) FARM, two-way mirroring.

(c) Non-FARM, RAID 5

(d) FARM, RAID 5

Figure 9. Distribution of data loss probability for a system under adaptive data redundancy schemes.

place disk drives in a concentrated way, data loss is about 12% more likely in ten-year period. During the replacement time (from the fifth year to the sixth year) data loss probability is 0.64% by gradual replacement, but 1.06% in concentrated replacement, due to the cohort effect. Once new disk drives are added to the system, data reorganization is done by using a fast algorithm [15]. We use a scheme that places no more than one replica out of two of any data object on the the newly-added disks. As a result, we do not observe much data loss occurrence after the replacement period, as Figure 7 shows. However, we found that the extra bookkeeping effort of age-based gradual replacement does not gain an advantage over random gradual replacement. This is good news for the management of large-scale storage systems.

5.4. Data Protection Based on Disk Drive Age

In addition to judicious replacement of drives, we can also adjust the redundancy to the risk inherent in the system. We considered two simple schemes. First, instead of simply mirroring objects when one object is stored on disk that is less than T months old, we kept three copies of the object. Second, we use $m+k$ redundancy unless one of the object groups is stored on a disk less than T months old. If this is the case, we add one additional parity object. In both schemes, if there is only one disk that is less than T months old and if the randomly selected new disk is older, then we merely replace the not yet burnt in disk with the older one.

In our first experiment, we simulated a system with 10,000 initially new disks, all using the failure rate predicted by our 4 state Markov model. We used replication and $m+k$ RAID 6 as our redundancy scheme. In the adaptive scheme, we switched from a more aggressive redundancy scheme (triplication, 4+2) to a less aggressive scheme (mirroring, 4+1 RAID 5) after six and twelve months respectively for a total six years of simulation time. We present the data loss probability over the six years under varied configurations (based on 20,000 data points for each scheme) in Table 4. We further plot the probability density function curves of data loss distribution to examine the infant mortality effect under the adaptive data redundancy schemes. As shown in Figure 9, we observe that the data loss probability when the storage system is young is greatly reduced by our adaptive data redundancy schemes as compared to the static redundancy scheme.

6. Conclusions and Future Work

Reliability of large disk based systems as well as precise modeling of disk drives still poses many unsolved problems. We have discussed the effect of disk infant mortality and have shown that it has an effect on storage systems with tens of thousands of disks. By comparing the results obtained with different disk models, we have shown that the exact failure rate curve is less important than the mere consideration of infant mortality. This opens the possibility to use analytical Markov models to calculate the reliability of systems of this size.

A few disk drives with a higher failure rate than normal do not cause damage in the large scale systems that are currently deployed or under development. Rather, system failure rate shoots up only if large numbers of disks have elevated failure rates. To avoid the "cohort effect," we suggest introducing new disks into the system in reasonably small batches—when disks need to be replaced because of their age, a gradual approach is noticeably better.

In addition to limiting the cohort effect, we can adjust the data protection scheme in response to its presence. We have shown that an adaptive data redundancy scheme that adds a stronger data protection for a batch of disk drives at their young ages will greatly reduce the high data loss rate brought by the effect of infant mortality. This suggests that a large-scale storage system has to manage redundancy autonomously, taking infant mortality into account, just as other systems such as AutoRAID [29] manage data storage autonomously based on data access patterns.

In future work, we will address the problem of efficient calculation. We will use this capability and simulation to investigate further efficient strategies of coping with variable disk failure rate, such as storage management based on disk age hierarchy. We will also study the effect of infant mortality on long-term system reliability.

Acknowledgments

Qin Xin and Ethan Miller were supported in part by Lawrence Livermore National Laboratory, Los Alamos National Laboratory, and Sandia National Laboratory under contract B520714. Thomas Schwarz was supported in part by an SCU-internal IBM research grant. Additional funding for the Storage Systems Research Center was provided by Engenio Information Technologies, Hewlett Packard, Hitachi Global Storage Technologies, IBM, Intel, Microsoft, Network Appliance, Rocksoft, Veritas, and Yahoo.

References

[1] A. Adya, W. J. Bolosky, M. Castro, R. Chaiken, G. Cermak, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. Wattenhofer. FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, Boston, MA, Dec. 2002. USENIX.

[2] D. Anderson, J. Dykes, and E. Riedel. More than an interface–SCSI vs. ATA. In *Proceedings of the Second USENIX Conference on File and Storage Technologies (FAST)*, San Francisco, CA, Mar. 2003.

[3] S. H. Baek, B. W. Kim, E. J. Joung, and C. W. Park. Reliability and performance of hierarchical RAID with multiple controllers. In *Proceedings of the Twentieth ACM Symposium on Principles of Distributed Computing (PODC 2001)*, pages 246–254. ACM, 2001.

[4] W. A. Burkhard and J. Menon. Disk array storage system reliability. In *Proceedings of the 23rd International Symposium on Fault-Tolerant Computing (FTCS '93)*, pages 432–441, June 1993.

[5] J. A. Carrasco. Validation of approximate dependability models of a RAID architecture with orthogonal organization. In *Proceedings of the 2003 International Conference on Dependable Systems and Networking (DSN 2003)*, pages 699–708, San Francisco, CA, June 2003.

[6] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-performance, reliable secondary storage. *ACM Computing Surveys*, 26(2), June 1994.

[7] A. Cohen and W. A. Burkhard. Segmented information dispersal (SID) data layouts for digital video servers. *IEEE Transactions on Knowledge and Data Engineering*, 13(4):593–606, July 2001.

[8] J. B. Dugan and G. Ciardo. Stochastic Petri net analysis of a replicated file system. *IEEE Transactions on Software Engineering*, 15(4):394–401, Apr. 1989.

[9] J. G. Elerath. Specifying reliability in the disk drive industry: No more MTBF's. In *Proceedings of 2000 Annual Reliability and Maintainability Symposium*, pages 194–199. IEEE, 2000.

[10] S. Frølund, A. Merchant, Y. Saito, S. Spence, and A. Veitch. FAB: Enterprise storage systems on a shoestring. In *Proceedings of the 9th Workshop on Hot*

Topics in Operating Systems (HotOS-IX), Kauai, HI, May 2003.

[11] S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google file system. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, Bolton Landing, NY, Oct. 2003. ACM.

[12] G. A. Gibson. *Redundant Disk Arrays: Reliable, Parallel Secondary Storage*. PhD thesis, University of California at Berkeley, 1990.

[13] G. A. Gibson and D. A. Patterson. Designing disk arrays for high reliability. *Journal of Parallel and Distributed Computing*, 17(1-2):4–27, 1993.

[14] A. Haeberlen, A. Mislove, and P. Druschel. Glacier: Highly durable, decentralized storage despite massive correlated failures. In *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation (NSDI '05)*, Boston, MA, May 2005.

[15] R. J. Honicky and E. L. Miller. Replication under scalable hashing: A family of algorithms for scalable decentralized data distribution. In *Proceedings of the 18th International Parallel & Distributed Processing Symposium (IPDPS 2004)*, Santa Fe, NM, Apr. 2004. IEEE.

[16] R. Y. Hou and Y. N. Patt. Using non-volatile storage to improve the reliability of RAID5 disk arrays. In *Proceedings of the 27th International Symposium on Fault-Tolerant Computing (FTCS '97)*, pages 206–215, 1997.

[17] IBM Corporation. IceCube – a system architecture for storage and Internet servers. http://www.almaden.ibm.com/StorageSystems/autonomic_storage/CIB_Hardware/.

[18] M. Kaâniche, L. Romano, Z. Kalbarczyk, R. Iyer, and R. Karcich. A hierarchical approach for dependability analysis of a commercial cache-based RAID storage architecture. In *Proceedings of the 28th International Symposium on Fault-Tolerant Computing (FTCS '98)*, pages 6–15, 1998.

[19] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An architecture for global-scale persistent storage. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Cambridge, MA, Nov. 2000. ACM.

[20] J. Y. B. Lee and R. W. T. Leung. Design and analysis of a fault-tolerant mechanism for a serverless video-on-demand system. In *Proceedings of the 9th International Conference on Parallel and Distributed Systems (ICPADS '02)*, pages 489–494, 2002.

[21] R. A. Meyer and R. Bagrodia. PARSEC user manual, release 1.1. http://pcl.cs.ucla.edu/projects/parsec/.

[22] S. W. Ng. Crosshatch disk array for improved reliability and performance. In *Proceedings of the 21st International Symposium on Computer Architecture*, pages 255–264, Chicago, IL, 1994. ACM.

[23] M. Schulze, G. Gibson, R. Katz, and D. Patterson. How reliable is a RAID? In *Proceedings of Compcon '89*, pages 118–123. IEEE, Mar. 1989.

[24] T. J. Schwarz. Generalized Reed Solomon codes for erasure correction in SDDS. In *Workshop on Distributed Data and Structures (WDAS 2002)*, Paris, Mar. 2002.

[25] S. Shah and J. G. Elerath. Disk drive vintage and its effect on reliability. In *Proceedings of 2004 Annual Reliability and Maintainability Symposium*, pages 163–165. IEEE, 2004.

[26] N. Talagala and D. Patterson. An analysis of error behaviour in a large storage system. In *Proceedings of the Workshop on Fault Tolerance in Parallel and Distributed Computing (FTPDC '99)*, 1999.

[27] The International Disk Drive Equipment & Materials Association (IDEMA). R2-98: Specification of hard disk drive reliability.

[28] H. Weatherspoon and J. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison. In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS 2002)*, Cambridge, Massachusetts, Mar. 2002.

[29] J. Wilkes, R. Golding, C. Staelin, and T. Sullivan. The HP AutoRAID hierarchical storage system. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles (SOSP '95)*, pages 96–108, Copper Mountain, CO, 1995. ACM Press.

[30] Q. Xin, E. L. Miller, T. J. Schwarz, D. D. E. Long, S. A. Brandt, and W. Litwin. Reliability mechanisms for very large storage systems. In *Proceedings of the 20th IEEE / 11th NASA Goddard Conference on Mass Storage Systems and Technologies*, pages 146–156, Apr. 2003.

[31] Q. Xin, E. L. Miller, and T. J. E. Schwarz. Evaluation of distributed recovery in large-scale storage systems. In *Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing (HPDC)*, pages 172–181, Honolulu, HI, June 2004.