Partial Disk Failures: Using Software to Analyze Physical Damage

Hai Huang
IBM TJ Watson Research
haih@us.ibm.com

Kang G. Shin
The University of Michigan
kgshin@eecs.umich.edu

Abstract

A good understanding of disk failures is crucial to ensure a reliable storage of data. There have been numerous studies characterizing disk failures under the common assumption that failed disks are generally unusable. Contrary to this assumption, partial disk failures are very common, e.g., caused by a head crash resulting in a small number of inaccessible disk sectors. Nevertheless, the damage can sometimes be catastrophic if the file system meta-data were among the affected sectors. As disk density rapidly increases, the likelihood of losing data also rises. This paper describes our experience in analyzing partial disk failures using the physical locations of damaged disk sectors to assess the extent and characteristics of the damage on disk platter surfaces. Based on our findings, we propose several fault-tolerance techniques to proactively guard against permanent data loss due to partial disk failures.

1. Introduction

High-capacity, low-cost magnetic disks are the building blocks of modern information storage and retrieval. It was estimated that over 90% of all new information produced in 2002 was stored on magnetic media [9] where most are hard disks. With current-generation disks' bit-density rapidly increasing, it is becoming much more difficult to maintain data integrity. Even if disk manufacturers can keep the *percentage* of random bit flips on the current-generation disks the same as older-generations, the fact that the current-generation disks are used to store and access a several orders-of-magnitude larger data volume means that they would have that much more number of bit corruptions. As each flipped or damaged bit may render a large portion of a disk (in the worst case, the entire disk) inaccessible, larger capacity disks are, therefore, more susceptible to data loss. Moreover, a head crash could cause significantly more damage to platters with higher density due to the more densely packed bits. The much larger volume of data that needs to be stored reliably presents a significant challenge to users and system administrators.

In enterprise environments, Redundant Array of Inexpensive Disks (RAID) [10] is extensively used to improve data fault-tolerance by storing additional parity information on multiple disks. However, due to form-factor constraints, it is not feasible to equip all platforms with multiple disks, e.g., laptops and workstations. Therefore, for single-disk machines, it is a common practice to back up data, manually or periodically, to optical media or external drives. However, such practice requires a strict self-discipline, and thus, is usually beyond the ability of most users and system administrators. As a result, data loss on single-disk machines is common, and it can be difficult both technically and financially to recover from it, if that is even possible. A simple keyword search on *data recovery* in Google resulted in close to 100 million hits, revealing many thousands of data-recovery companies in this space.

One possible solution for providing better fault-tolerance for data is proposed by the Free Space File System (FS2) [7]. FS2 exploits unused disk capacity for storing replicas of data blocks, and by placing the related replicas physically adjacent to one another, disk I/O access time and energy efficiency are significantly improved. As a side effect, the replicas can also provide some protection to the data they mirror. Unfortunately, replication decisions made by FS2 is optimized for performance and not for fault-tolerance. For example, FS2 does not give additional weight to file system meta-data over data when replicating, which is intuitively appealing for data fault-tolerance as a corrupted or failed meta-data block can potentially cause much more damage than an inaccessible data block. FS2 also does not differentiate among data blocks, but some files (e.g., those in /home and /etc) are much more important for safe-keeping than others (e.g., in /tmp). Replication priority, most of the time, can be intuitively defined (e.g., meta-data over data, /etc over /tmp), and user-defined policies [18] can be used for other data.

An equally, if not more, critical decision is at which location replicas should be placed on disk. This location (with respect to the location of the original data) can sig-

24th IEEE Conference on Mass Storage Systems and Technologies (MSST 2007)
0-7695-3025-7/07 $25.00 © 2007 IEEE

nificantly impact both the disk's performance and fault-tolerance. For fault-tolerance, one would like to keep replicas as *physically* far away as possible from their original data, and from one another, in such as way that the probability they are simultaneously damaged/destroyed (which would result in permanent data loss) is minimized. For performance, however, keeping replicas *physically* far away from their original data could cause long mechanical (seek and rotational) delays when the original data is updated. Even if we could minimize the performance impact by relaxing the data-consistency guarantee using asynchronous disk IOs for replica updates, it is not clear how to place the replicas *physically* far away from their original data given that the physical geometry of modern disks is often hidden. Logical distance between two sectors is a poor indicator of their physical distance. For example, two sectors of 1,000,000 sectors apart (logical distance) can be located on different zones and different surfaces (physical distance) on one disk, but located in the same zone and on the same surface on another disk. This also applies for two pairs of sectors on the same disk, with sectors of each pair being the same logical distance apart but located at different regions on the disk.

Therefore, to most effectively make use of replication for fault-tolerance while keeping performance impact low, one would need to have detailed information of a disk's physical geometry, e.g., sectors per track, tracks per zone, number of cylinders, surfaces, and platters. The physical features, such as tracks, surfaces, platters, can be used to create physical barriers between the original data and its replicas to minimize the likelihood of a permanent data loss. Our contributions in this paper are listed as follows.

- We develop a methodology of doing post-mortem analysis of partial disk failures for gaining a better understanding how data on disk get lost or corrupted. This knowledge allows us to decide whether placing a replica on a different platter, or a different surface, or a different track, or a different zone from the location of its original data is the best way to improve fault-tolerance. We show that for disks of different manufacturers or the same manufacturer but different generations, the best replica placement strategy is often different.

- We develop a set of software tools to quickly and accurately extract the physical geometry of disks. This extraction process can be performed either offline or online. Disk manufacturers often withhold this information, and as we will see later, in case even if this information is disclosed, the disclosed specifications are too coarse to be useful. Detailed information of disk's physical geometry is critical to replica placement. For example, if we know that the probability of disk blocks located on different platters being simultaneously destroyed is extremely low for a particular disk model (from the failure analysis described above), precise physical geometry is need to avoid placing replicas on the same platter as their original data.

The rest of this paper is organized as follows. Section 2 very briefly gives some background information on the anatomy of modern magnetic disks. Section 3 describes the probing algorithm for discovering disks' physical geometry. Section 4 describes our study of 9 partially failed hard drives. Section 5 covers the related works, and Section 6 concludes this paper.

2. Anatomy of Magnetic Disk

Disk drives contain both mechanical and electronic components. The mechanical component consists of recording (i.e., disk platters and read/write heads) and positioning parts (i.e., an arm assembly that moves disk heads into the correct position and a track-following system that keeps it in place). The controller portion is composed of a microprocessor, cache memory, and an I/O bus interface. The disk controller manages the storage and retrieval of data to and from the disk and performs mappings between the logical address requested by the CPU or DMA controller and the physical disk location. We now detail both the mechanical components and the controller components in the following sections.

2.1. The Mechanical Component

Magnetically-coated disk platters are the most crucial parts of a disk drive as all data are encoded on their surfaces. Disk platters rotate in a lock-step fashion at a constant angular velocity, and as the disk head flies directly above a disk region, data stored in that region can be read and written by detecting and altering magnetic flux variations on the platter surface.

Data are organized in circular tracks on recording surfaces. Since the inner region of a recording surface has shorter tracks than those on the outer region, surfaces are partitioned into multiple zones to more efficiently store data without overly complicating the recording and positioning mechanics. A zone contains consecutively laid-out tracks, each with the same number of sectors. When there are multiple recording surfaces in a disk, tracks (one per surface) that are vertically aligned with each other, form a cylinder. Data stored on tracks of the same cylinder can be accessed quickly from one to another as there is no mechanical movement of disk heads. If a replica is stored

0-7695-3025-7/07 $25.00

on the same cylinder as its original data but on a different track (surface), not only the two disk blocks can be accessed quickly from one another, being on different recording surfaces naturally protects the two data blocks (in most cases) from simultaneously destroyed as each surface has an independent disk head. We will show more evidence of this in Section 4.

Due to an average annual growth rate of 60% in recording density since early 90s, bits are encoded extremely densely on modern disks—hundreds of gigabits per square inch is becoming a standard. Several disk manufacturers are now beginning to push for the terabit range. However, this rapid increase in recording density (weaker magnetic flux) poses more strict timing requirements for the disk heads. The weaker magnetic flux can only be compensated by lowering the disk flying height, which increases the probability of disk head making contact with its recording surface: a *head crash*! Moderns disks use more rigid arm assembly and vibration sensors to minimize such occurrence, and more scratch-resistant surface lubricant, overcoat, and platter materials to reduce the amount of damage if such event does happen. Despite these efforts, head crashes and data loss are still common. *Minor* head crashes that temporarily over-heat a disk head are events that are even expected to occur from time to time and are compensated in most modern disk drives. However, when disks are spinning at 7200–15000 RPM, even a slight change in the environment can cause a minor crash to take a turn for the worst.

2.2. The Controller Component

Disk controller mediates access to the mechanical component, runs the track-following system, transfers data between the disk and its clients, and, in many cases, manages an embedded cache. Controllers are built around specially-designed embedded processors, which often have digital signal processing capability and special interfaces that allow them to control hardware directly. There has been a trend toward more powerful controllers for handling increasingly-sophisticated interfaces and for reducing costs by replacing previously-dedicated electronic components with firmware.

2.3. Fault-Tolerance

Ideally, we would like our storage components to have 100% availability and never fail. In reality, however, hard disk failures are fairly common. The two most commonly-occurring types of failure in hard disks are electronic and mechanical failures. An electronic failure occurs when the controller board on the disk fails, commonly caused by a transient electrical surge. Fortunately, data are often recoverable after this type of failures as platters are not usually damaged in the process. Recovery can be as simple as replacing the controller board from an identical disk. Mechanical failures, on the other hand, have more serious consequences. Common causes include mechanical shocks, contamination, condensation, servo errors, and head crashes (cause approximately 45% of the cases where data is lost [1]). Mechanical failures occur more frequently than we would like, and without a proper means to guard against them, all data stored on a disk can be destroyed beyond recovery in a few seconds.

3. Probing for Physical Geometry

To accurately characterize how failed sectors are formed on platter surfaces so that we can better place data and their replicas, the disk's physical geometry must be known. This is used to map the logical address of a failed sector to its residing track, zone, recording surface, and platter. Despite manufacturers' unwillingness to disclose disk's physical geometry, some physical attributes are always published—disk rotation speed, number of platters and recording surfaces, etc. For our purpose, the only other information we need to know is the number of sectors per track, from which we can derive other needed parameters. For example, the number of zones and tracks per zone can be derived from the sectors-per-track information as tracks from different zones have a different number of sectors per track.

There are several previously-proposed techniques [2, 5, 13, 16, 17] on extracting disk's physical features. Worthington and Schindler [13, 17] relied on interrogative SCSI commands for extraction, but this technique is not applicable to IDE disks. Experimental methods using empirical observations were first proposed by Aboutabl *et al.* [2] and later improved by Talagala *et al.* [16] and Dimitrijevic *et al.* [5].

The probing algorithm propose here is also based on observing empirical results. However, unlike the previous approaches with their main purpose being extracting as much physical geometry information as possible (even though some of this information is already known), we only need to extract the sectors-per-track information, but as quickly as possible and incur as few number of I/O operations as possible. We need to perform this extraction process quickly if the extraction happens offline or to incur a small number of I/O operations if it is done online. Our algorithm meets these requirements well in both modes and is presented in Figure 1.

This algorithm is based on the technique proposed by Dimitrijevic [5]—sequentially write to the disk, two sectors at a time, and compare its access time with the access time of the two previously-accessed sectors. If the difference between the two access times is greater than a certain thresh-

```
#Definitions
DEFINE  SECTOR_SIZE             512
DEFINE  SETTLE_TIME             500
DEFINE  BACKTRACK_SECTORS       5
DEFINE  REQUIRED_CONFIRMS       10
DEFINE  FULL_ROTATION           8333


# Initialization
i_save          = 0;
confirmed       = 0;
warp_location   = 0;
warp_multiplier = 1;
last_spt        = 0xbeefbeef;
last_atime      = FULL_ROTATION;


FOR (i = 0; i <= end_sector; i++)
    # Seek to the next sector and write 2 sectors
    lseek(i * SECTOR_SIZE);
    gettimeofday(&t1);
    write(2 * SECTOR_SIZE);
    gettimeofday(&t2);
    atime = t2 − t1;
    diff = abs(atime − last_atime);
    # True when this might be a track/cylinder boundary
    IF ((diff > SETTLE_TIME) AND
        (i_saved = i OR confirmed = 0))
        IF (++confirmed = REQUIRED_CONFIRMS)
            # We have confirmed multiple times this is a track/cylinder
            # boundary and we modify warp_multiplier to skip ahead
            IF (warp_multiplier > 1)
                IF ((i − warp_location + 1) MOD last_spt = 0)
                    warp_multiplier *= 2;
                ELSE
                    # Warped to a wrong location so we go back
                    warp_multiplier = 1;
                    i = warp_location + last_spt − BACKTRACK_SECTORS;
                    continue;
            ELSE
                IF (i − warp_location + 1 == last_spt)
                    warp_multiplier = 2;
                ELSE last_spt = i − warp_location + 1;
            # From warp_location to i, sectors_per_track is last_spt
            warp_location = i + 1;
            confirmed = 0;
            i += last_spt * warp_multiplier − BACKTRACK_SECTORS;
        ELSE
            # We backtrack multiple times to make sure that sector i
            # is really a track/cylinder boundary
            i_saved = i;
            backtrack_sectors = BACKTRACK_SECTORS;
            i −= BACKTRACK_SECTORS;
    # Sector i does not seem to be a track/cylinder boundary
    ELSE
        IF (confirmed && −−backtrack_sectors == 0)
            # After backtracked, we failed to confirm
            confirmed = 0;
        last_atime = atime;
```

Figure 1. The proposed algorithm that quickly and accurately probes for sectors-per-track information. Some boundary cases are not included to make the algorithm more succinct and readable.

old, it is marked as a potential cylinder or a track boundary. In our algorithm, when we find a potential boundary, we try to backtrack and re-probe multiple times to ensure that the boundary found is really a boundary and not something transient. We found this extra check is very important for IDE disks where transient behaviors are fairly common. By backtracking and rewriting multiple times, we can eliminate most inaccuracies from our probing results.

More interestingly, to speed up the probing process and minimize the number of I/O operations, we take advantage of the fact that there is a very high probability that any subsequent tracks will have the same number of sectors on them as their previous track. This allows us *skipping ahead* by hundreds of sectors and probe only the disk region where we think the next boundary is located. If we have guessed these boundaries correctly multiple times, we will start *warping ahead* in multiples of tracks, which can more significantly speed up the probing process. When we guessed correctly, especially in the latter case, which did happen often in our experiments, we only needed a few disk accesses to probe hundreds of tracks as opposed to hundreds of accesses for a single track. If we guessed wrong, we can always backtrack. This does happen sometimes (due to slip sectors and spare sectors, which we will discuss later), but even if it does happen, the overhead of making the necessary correction is fairly small—tens of milliseconds. A sample of the extracted sectors-per-track information for an IBM Ultrastar disk is shown in Table 1.

Sectors	Tracks	Sectors-per-track
0–503	1	504 per track
504–1007	1	504 per track
1008–2015	2	504 per track
2016–4031	4	504 per track
4032–8063	8	504 per track
8064–16127	16	504 per track
16128–32255	32	504 per track
32256–64511	64	504 per track
...		

Table 1. Sectors-per-track information obtained from an IBM Ultrastar disk. From sector 0 to 503, we write 2 sectors at a time and find that sector 503 is a track boundary. We then make the assumption that the following track also have 504 sectors. Instead of writing sequentially, we write only to the assumed boundary region. If we guessed correctly, we can warp ahead and quickly finish the probing process.

The sectors-per-track information we extracted from the entire disk is shown in Figure 2. From this figure, we can

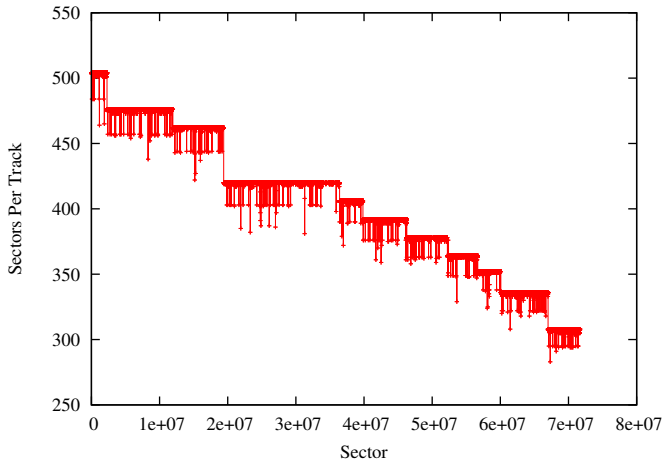

Figure 2. Sectors-per-track information of the entire IBM Ultrastar disk.

clearly see that the disk has 11 zones. On each zone, most of the tracks have the same number of sectors, with some variations. These variations are caused by slip sectors and spare sectors. Slip sectors are used to skip physical defects on media surfaces created during manufacturing, and they reduce the number of usable sectors on the affected tracks. Spare sectors are used to remap bad sectors created at runtime. In case a good sector becomes defective, disk firmware will automatically remap the defective sector to one of the nearby spare sectors. Spare sectors also cause the number of usable sectors to decrease on the tracks where they are pre-allocated. As we will see in the next section, these variations are important to take into account when characterizing disk failures.

4. Case Studies

To study partially-failed disks at a microscopic level as we are doing here, privacy is a major issue. Unlike previous works in which the focus is mostly on failure rates and their correlation with disks' age, manufacturer, workload, etc., where data-mining the hardware inventory list often suffices, we, on the other hand, need to examine all disk sectors post-mortem, in detail. As most data can be recovered partially or even completely from these disks, obtaining failed disks for our purpose was a challenge by itself. After jumping over some red tapes, having made promises to safely and utterly dispose all contents stored on the failed disks (by drilling nails into them), and signing the required paperwork, we finally received 60 failed disks—10 from the University of Michigan's EECS Department's Computing Organization (DCO) and another 50 from the University of Michigan's Property Disposition (PD) Division.

4.1. Overview

Most of the failed disks (shown in Figure 3) were produced by large disk manufacturers, i.e., Maxtor, Western Digital, Seagate, Quantum, and IBM, simply because these companies have or have had a much larger market share than others. However, the relative number of these disks does not accurately reflect the manufacturers' true market share as (i) the sample size used in our study is small, and (ii) disks made by some companies are more or less prone to failures than others.

We categorize these failed disks into three groups—non-bootable disks, disks with bad sectors, and perfectly healthy disks. We now describe each of the three categories in more detail.

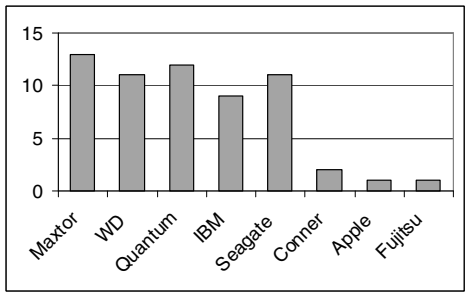

Figure 3. Categorizing the 60 failed disks by their manufacturers.

- **Nonbootable:** These are disks cannot be recognized by BIOS at the power-on time, and they account for more than half of the failed disks (37). This observation initially led us to believe that most disks would become useless when failure occurs. However, after studying these disks in more detail, we negated this hypothesis. Instead, we believe these disks are now non-bootable has little to do with why they were taken out of operation in the first place. We found all 37 non-bootable disks came from the batch we obtained from PD, where most have significant external physical damages, e.g., dents and broken interface pins. However, having external damages on internal disks is highly unusual as internal disks are always physically protected by sturdy external computer cases. Instead, these external damages are most likely incurred after they were replaced and during the handling process after they arrived at PD. On these disks, as the state of data cannot be accessed without using special hardware and a clean room, we could not assess the extent of damage on their recording surfaces to characterize their failure state. Therefore, we exclude this type of failures from our analysis.

- **Badsectors:** These are bootable disks (9) with at least one defective sector. Having one defective sector out of billions of sectors on a disk is not that uncommon. However, losing 512 bits of information can either be harmless if the affected sector is not currently mapped to any data, or it can be very destructive if the affected sector contains important file system metadata. A sector can become corrupted for various reasons: material degeneration (caused by overuse or old age) or a head crash. By identifying and mapping the exact physical location of these defective sectors, we hope to gain a better understanding of how data become corrupted to better safeguard data.

- **Healthy:** Ironically, some "failed" disks (14) are actually perfectly healthy, but for some reason, were identified as defective by the system administrator. These disks have no bad sectors nor any anomalies in their SMART status. SMART (Self-Monitoring Analysis and Reporting Technology) monitors mechanical and electronic components of a disk and gives advance warnings to predictable failures. There are several possibilities that these healthy disks were falsely identified as broken. One possibility is bit-flipping, which happens approximately once for every 10^{14} bits accessed. Perhaps, disk scrubbing [15] can alleviate such problems. Another possibility is bad writes—sometime a write operation writes to a wrong location or in between two tracks. Besides these hardware errors, bugs in the file system and human errors could also have caused a healthy disk to be falsely identified. But as most production grade file systems are already very stable and system management software are becoming more commonly deployed to minimize human errors, we believe concentrating on disk hardware errors is more fruitful and beneficial. Even though there is no way for us to assess the extent of damage on these disks, using replication will also help alleviate this type of failures.

Our focus is to analyze failed disks with defective sectors. This characterization will help us better place data and their replicas on disk to minimize the possiblity of permanent data loss. Among the 9 disks with defective sectors, 2 are IBM, 2 are Western Digital, 2 are Quantum, 2 are Maxtor, and 1 is Seagate. The Seagate disk has only 1 defective sector and is not interesting to discuss, and therefore, is not included in our discussion. We will now characterize and dissect the extent of damage on the 8 remaining disks.

4.2. IBM Deskstar 75GXP

The specification of the IBM Deskstar 75GXP disk is shown in Table 2, listing its capacity, revolutions per second (RPM), interface type, number of zones, cylinders, platters, and recording surfaces. This information (except zones) is generally published in the data specification sheet of today's disks and does not require probing. By using commercial software to scan the disk's recording surfaces, we found a list of defective sectors and their logical address.

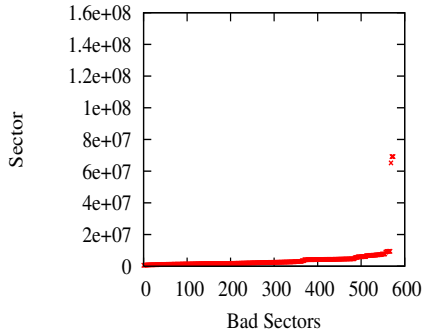

Figure 4. IBM 75GXP: Locations of bad sectors.

Capacity	75 GB
Zones	15
Cylinders	27724
Platters	5
Surfaces	10
RPM	7200
Interface	IDE

Table 2. IBM 75GXP: Disk specification.

Figure 4 shows a list of 575 bad sectors found on this disk. Almost all of them are found in only low-address regions on the disk, which indicates that by simply putting some distance between replicas and their original data block by a reasonably large number of sectors (logical distance), replicas can be used effectively to improve fault-tolerance. This matches the conventional wisdom well, and it is also the most straightforward way to improve fault tolerance. It is, however, most likely not the most efficient way of placing replicas. As we have pointed out earlier, placing replicas far away from their original data can significantly hurt performance. From this analysis, we hope to be able to place replicas closer to their original data (to minimize performance impact in keeping replicas in sync) while providing the same level of fault-tolerance.

Table 3 shows the published physical geometry of 75GXP, which we used to perform logical to physical address mapping. For each defective sector, this gives us its exact physical location in terms of which platter, surface, zone, cylinder, and track the defective sector resides. First, we mapped the defective sectors to platter surfaces as shown in Figure 5. This figure indicates that surface #7

has suffered the most amount of damage with 352 bad sectors. Other damaged sectors are scattered across surfaces #2, #3, #4, #5, #6, #8, and #9, indicating that bad sectors exhibit very poor physical locality. This made little sense as physical damages tend to co-locate, e.g., head crash causing nearby sectors at the point of contact to become inaccessible, or a previously-damaged region is damaged again due to particle built-up in that region.

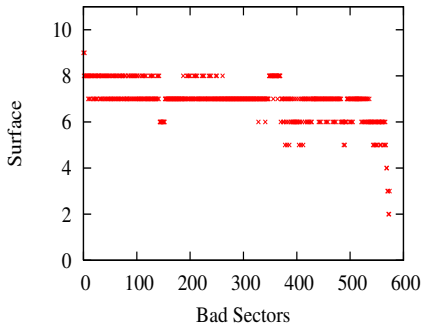

Figure 5. IBM 75GXP: A mapping of bad sectors to recording surfaces using the disk's published physical geometry.

As it turned out, using the published physical geometry is not sufficient to derive an accurate mapping between logical address and physical address. The actual disk geometry, which vary from disk to disk even for disks of the same manufacturer and same model, is slightly different from the published one. However, this slight variance makes a big difference in the mapping of the defective sectors. The difference between the actual geometry and the published one is shown in Figure 6, and this difference is caused by the inevitable omission of slip sectors and spare sectors in the published specification, which is understandable, as different disks have different manufacturing flaws. Using the actual physical geometry and the probing method described in the previous section, we re-mapped the defective sectors to recording surfaces. This result is shown in Figure 7.

Zones	Cyl start	Cyl end	Sectors/Track
0	0	1375	702
1	1376	2831	684
2	2832	4239	666
3	4240	6975	648
4	6976	9759	612
5	9760	11551	594
6	11552	13631	567
7	13632	16239	540
8	16240	18319	504
9	18320	19567	486
10	19568	21199	459
11	21200	23519	432
12	23520	25215	396
13	25216	26319	378
14	26320	27724	351

Table 3. IBM 75GXP: disk's published physical geometry.

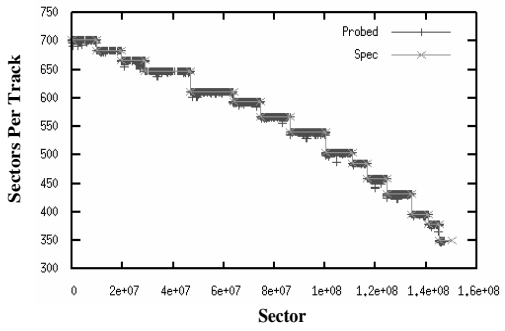

Figure 6. IBM 75GXP: Comparing the *actual* physical geometry (Probed) of the disk with its published data (Spec).

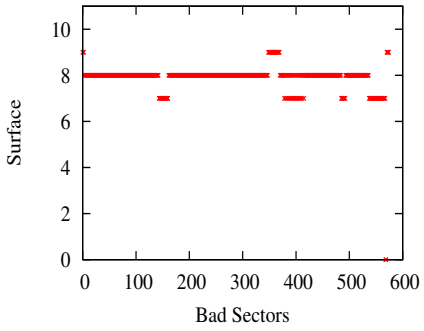

Figure 7. IBM 75GXP: A mapping of bad sectors to recording surfaces using the actual physical geometry of the disk.

Using the actual disk geometry, we found that surface #8 has the most number of damaged sectors (467). Other defective sectors, instead of being scattered across many different recording surfaces, are now localized to only the two neighboring surfaces #7 and #9. This finding suggests that it is not necessary to have replicas placed far away from their original data to benefit from replication. *It is sometimes sufficiently safe to place the replicas on the same cylinder as the original data but on a different recording surface (not on a neighboring surface though).* This placement strategy allows us to store replicas in such a way that replication provides as much fault-tolerance benefits as the traditional method, i.e., placing replicas far away from their original data, yet, keeps performance overhead to a minimum when synchronizing replicas with the original data.

Sector	Track
813160	1158
819801	1168
827144	1178
833786	1188
841128	1198
847770	1208
855112	1218
862455	1229
869096	1238
876439	1248
883080	1258
890423	1268
897064	1278
904407	1288
911048	1298
918391	1308
925032	1318
932375	1328
939017	1338
946359	1348
953702	1359
960343	1368
967686	1378
974327	1388
...	...

10 tracks
10 tracks
10 tracks
...

Table 4. IBM 75GXP: A sample mapping of bad sectors to their track.

Next, we mapped the defective sectors to tracks, and a sample of this mapping is shown in Table 4. A closer examination of this table shows a pattern—many of these sectors are 10 tracks apart from each other. The fact that this disk has 5 platters and 10 recording surfaces implies these sectors are on neighboring tracks of the same surface. Plotting their sector offset (within a track), as shown in Figure 8, we can see there is yet another clear pattern. We immediately suspected the regular pattern in the sector offsets is an artifact of head skews and cylinder skews. It might be an indication that these defective sectors are located physically adjacent to one another on neighboring tracks, e.g., as parts of a scratch or of a hole on a surface.

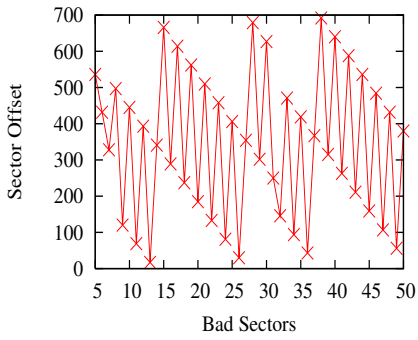

Figure 8. IBM 75GXP: A sample of sector offsets of the defective sectors.

To verify our hypothesis is correct, we first obtained the disk's head skew time (1.2 ms) and cylinder skew time (1.7 ms) from the disk's specification sheet. Head skew (HS) is the offset distance with respect to the start sector of the previous track in the same cylinder and is used to compensate for rotation delay when switching from one disk head to another. Similarly, cylinder skew (CS) is the offset distance from the start of the last track of the previous cylinder. Instead of using time unit to express these skews, we convert the time unit to number of sectors that would pass under the disk head for that amount of time, as shown below.

$$HS_{sectors} = \frac{HS_{time}}{Full\_Rotation_{time}} \times SPT$$

$$CS_{sectors} = \frac{CS_{time}}{Full\_Rotation_{time}} \times SPT.$$

A full rotation takes 8.33 ms on this disk (7200 RPM), and SPT stands for sectors per track. HS and CS are found to be 102 and 144 sectors, respectively. Next, we calculate the skewness of neighboring tracks on the same surface, called *surface skew*(SS), as shown below.

$$SS_{sectors} = (HS_{sectors} \times (Surfaces - 1) + CS_{sectors}) \% SPT.$$

The SS of this disk is approximately 360 sectors. This means if two sectors, on neighboring tracks of the same surface, have an offset difference of 360 sectors, they will fall on the same radius line passing through the center of the disk. An example illustrating different variations of this scenario is shown in Figure 9.

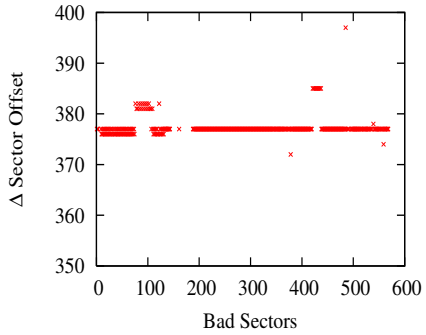

Figure 10. IBM 75GXP: Difference between bad sectors' sector offsets.

Differences in the sector offset between each pair of defective sectors that are on neighboring tracks are shown in Figure 10. These offset differences lie consistently around 375–380 sectors—a strong indication of a (linear) surface scratch, as discussed earlier. This might be caused by a

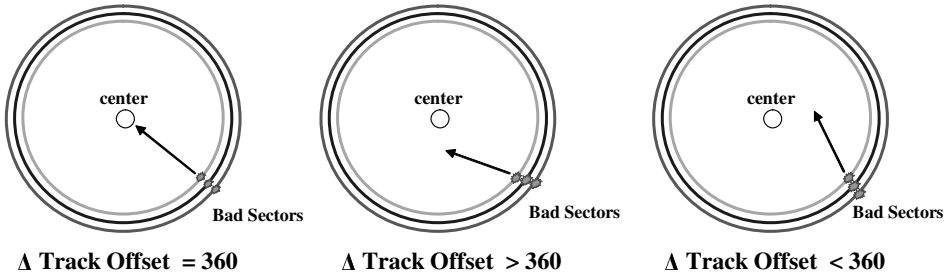

Figure 9. Using sector offset difference to predict sectors' relative location.

head crash as a head will leave a regular scratch pattern on the recording surface when it crashed. This analysis of sector offsets allows us to propose an alternative, and potentially better, method to replicate data—placing replicas on a physically adjacent track (on the same surface) as the original data, if the sector offset information is taken into account when making data placement decisions. With this placement strategy, an update operation that synchronously modifies a data block and its replicas can be completed with a very small amount of mechanical delays. With inter-track distance becoming so minuscule on modern disks, head often can just very quickly "re-settle" itself to move to a neighboring track. This mode of replication allows replicas to be placed very close to their original data while data are kept as fault tolerant as the previous method.

Capacity	36.7 GB
Zones	11
Cylinders	15110
Platters	6
Surfaces	12
RPM	10000
Interface	SCSI

Zones	Cyl start	Cyl end	Sectors/Track
0	0	378	504
1	379	2069	476
2	2070	3416	462
3	3417	6788	420
4	6789	7493	406
5	7494	8863	392
6	8864	10167	378
7	10168	11195	364
8	11196	11999	352
9	12000	13714	336
10	13715	15109	308

Table 5. IBM 36LZX: Disk specification.

4.2.1. Discussion On this IBM disk, all the defective sectors are found on zone #0. This might have been caused by either a head crash or an overuse of this disk region. Some file systems, e.g., NTFS, try to place more frequently-used files on the outer edge (low-address zones) of the disk to achieve higher I/O bandwidth. However, this causes some parts of the disk to be used much more frequently than others, and the excessive wear can cause damage over time. A file system that makes more balanced use of all disk regions, e.g., Ext2, can alleviate such problems.

The failure analysis that we have done for the IBM 75GXP disk suggests the existence of better strategies in placing replicas than the conventional method, which is to place replicas far away from each other and their original data. By placing replicas on a neighboring track (of the same cylinder or the same surface) of the original data, we can achieve the same level of fault-tolerance as the conventional method but with much less performance overhead in maintaining replicas. We are not suggesting drawing conclusion based on a single failure analysis, but rather that failure analysis can reveal interesting properties of physical disks that file system designer can use to better store data. In the following subsections, we will analyze more failed disks and show common failure characteristics among the disks of the same manufacturer and across different manufacturers.

4.3. IBM Ultrastar 36LZX

The second failed IBM disk is an Ultrastar 36LZX. Its specification is shown in Table 5. We found 3 times more defective sectors (1799) on this disk than the first IBM disk. Similar to the first IBM disk, all the defective sectors are found on zone #0 (shown in Figure 11), which, as we have suggested previously, might have been caused by an overuse of that region. The defective sectors found on the 36LZX, similar to the 75GXP, also have a heavy concentration on one surface (shown in Figure 12). The defective sectors not found on this surface are all found on the two neighboring surfaces as that we have seen on the first IBM

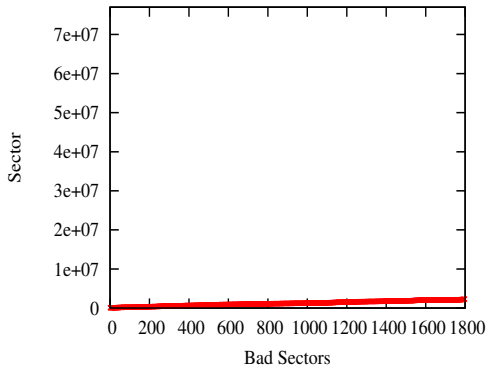

Figure 11. IBM 36LZX: Locations of bad sectors.

disk. We note that the most damaged recording surface is the inner surface of the outer-most platter on both disks.

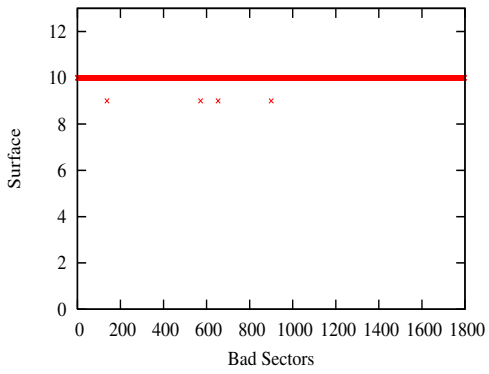

Figure 12. IBM 36LZX: Mapping of bad sectors to recording surfaces using the actual physical geometry of the disk.

It is not clear whether or not these similarities are failure characteristics of IBM disks as our sample size is too small to make such a general statement. These common failure characteristics would not be apparent if only logical addresses of the failed sectors are used.

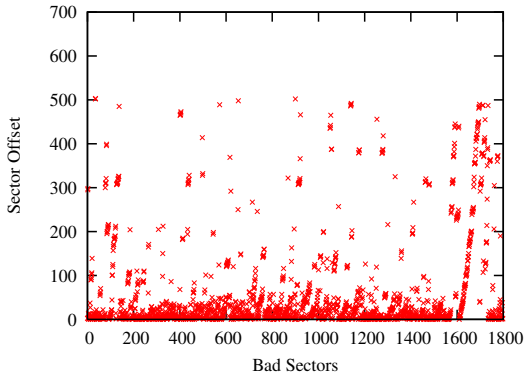

Figure 13. IBM 36LZX: Sector offsets of the defective sectors.

Despite the many similarities in the failure characteristics between the two disks, there are also some differences. Different from the first disk, where the regular patterns in sector offsets indicated a possible scratch on a recording surface, no such patterns are found on the second disk. Instead, we found offsets of defective sectors on the second disk as shown in Figure 13. This figure clearly show that most of the defective sectors have small offset values. A histogram of all the defective sectors by their offset value is shown in Figure 14. It is not clear what might have caused

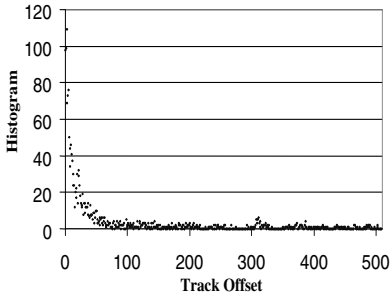

Figure 14. IBM 36LZX: A histogram of sector offsets—smaller ones dominate.

such a problem, but it might have been caused by the servo system malfunctioning.

4.3.1. Discussion The two IBM disks have several common failure characteristics. For example, almost all defective sectors are located on the outer-most zone and one recording surface. From these observations, we can design very efficient replica placement policies, e.g., store replicas and their original on different surfaces and avoid overusing a single zone. However, the differences in their failure characteristics are also important to take into account. The failure on the first disk says that, to protect against data loss from surface scratches (caused by a head crash), we need to take into account of sector offset if we were to place a data block and its replica on two neighboring tracks of the same recording surface. However, the failure on the second disk tells us that to protect against more severe destruction of data (e.g., servo problems), storing replicas on neighboring tracks of the same surface is still risky.

4.4. Western Digital Caviar 205BA

The first Western Digital (WD) disk we studied is a Caviar 205BA. Its specification is shown in Table 6. One significant difference between WD and IBM disks in their physical geometries is that WD disks have more zones than IBM disks.

Capacity	20.5 GB
Zones	20
Platters	3
Surfaces	6
RPM	7200
Interface	IDE

Table 6. WD 205BA: Disk specification.

We found 2835 defective sectors on this disk, shown in Figure 15. Among them, 2832 are consecutive, spanning consecutive tracks and multiple recording surfaces, as shown in Figure 16. This behavior is very different from IBM disks, and therefore, replication policies that work

well for IBM disks will not be effective on WD disks. On this disk, even though defective sectors have long runs, but in terms of affected tracks and cylinders, only a few are affected. From this observation, placing replicas only a few cylinders away from their original data seems to be a good simple solution.

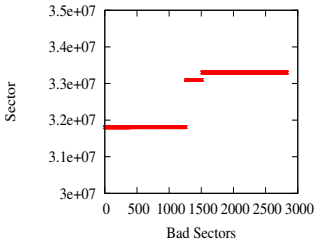

Figure 15. WD 205BA: Locations of bad sectors.

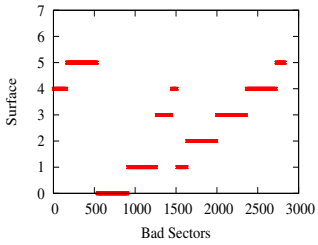

Figure 16. WD 205BA: Mapping of bad sectors to recording surfaces.

4.5. Western Digital Caviar WD400BB

The WD400BB disk is another WD disk but of a different generation from the WD205BA (WD400BB's recording density is much higher). Its specification is given in Table 7. Like the WD205BA, this disk also has more zones than the IBM disks.

Capacity	40 GB
Zones	18
Platters	2
Surfaces	4
RPM	7200
Interface	IDE

Table 7. WD 400BB: disk specification.

We found a list of 1069 defective sectors on this disk. More than half of them (692) are consecutive, which is similar to those found on the WD205BA, but the damage is not as extensive—consecutive sectors found on WD205BA span all surfaces whereas those on WD400BB span only 2 (shown in Figure 18) surfaces. As shown in Figure 17, almost all the consecutive sectors are near sector address #7.5e7.

Other than these consecutive failed sectors, there is another disk region which appears to be damaged in a different way (potentially caused by a separate failure). This is the region of the disk consists of 4 smaller failure regions shown in the beginning part of Figure 17. A closer examination of these defective sectors shows a pattern in their physical locations. We first mapped these sectors to tracks and surfaces, and a small sample of this mapping is shown in Table 8. From this mapping, it is clear that

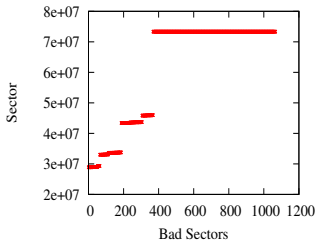

Figure 17. WD 400BB: Locations of bad sectors.

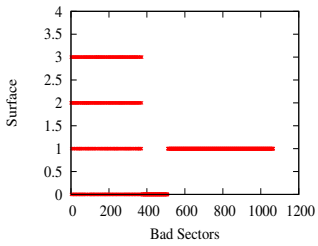

Figure 18. WD 400BB: Mapping of bad sectors to recording surfaces.

Sector	Track	Surface
28888401	31438	2
28889142	31439	3
28889882	31440	0
28890623	31441	1
28891363	31442	2
28892987	31443	3
28893728	31444	0
28894468	31445	1
28895209	31446	2
28895949	31447	3
28896690	31448	0
28898314	31449	1
28899054	31450	2
28899795	31451	3
28900535	31452	0
28901276	31453	1
28902016	31454	2
28902757	31455	3
28904381	31456	0
28944372	31502	2
28945996	31503	3
28946736	31504	0

Table 8. WD 400BB: A sample mapping of bad sectors to their track and surface.

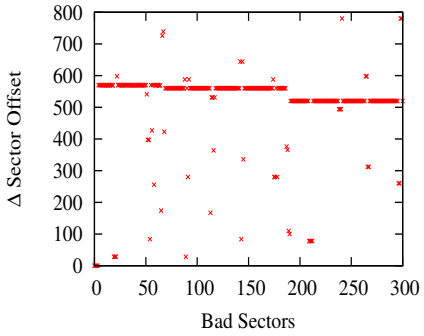

Figure 19. WD 400BB: Sector offset differences of bad sectors.

0-7695-3025-7/07 $25.00

the defective sectors in this disk region are all exactly one track apart. This indicates that on each surface, defective sectors in this particular disk region are located on neighboring tracks. Analyzing offsets of these sectors as we have done in 75GXP reveals consistent sector offset differences as shown in Figure 19, which is an indication of a surface scratch. Unlike 75GXP, where there was only one scratch, WD400BB appears to have multiple scratches, on every surface. This is possibly the result of one disk head crashing causes other heads to also crash from the vibration of the first crash.

4.6. Maxtor D540X

The two failed Maxtor disks are both D540X model, and both are covered in this subsection. These disks are older, low-end models compared to the other disks, with a single platter, a single recording surface, and an rotational speed of 5400 RPM. Their specification is shown in Table 9.

Capacity	20 GB
Zones	15
Platters	1
Surfaces	1
RPM	5400
Interface	IDE

Table 9. Maxtor D540X: Disk specification.

The defective sectors found on the two disks are shown in Figure 20 and Figure 21. Similar to the WD disks, bad sectors on the Maxtor disks also tend to locate either adjacently or within the same track. On one disk, 63 of the 84 defective sectors are co-located with at least one other defective sector on the same track. On the other disk, 93 of the 214 defective sectors are co-located in the same way. This is very similar to the failure characteristics of the WD disks. Given that the Maxtor disks, like the WD disks, also have significantly more zones than the IBM disks, the similarity in how Maxtor and WD disks place data on recording surfaces might explain the similarity in their failure characteristics.

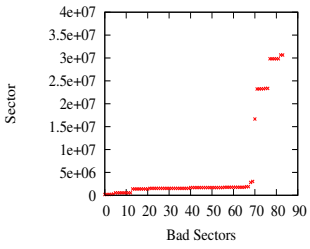

Figure 20. WD D540X 1: Locations of bad sectors.

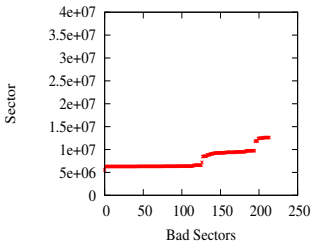

Figure 21. WD D540X 2: Locations of bad sectors.

4.7. Quantum Fireball LCT10

Quantum Fireball LCT10's specification is given in Table 10. The defective sectors found on this disk are shown in Figure 22, and the mapping to surfaces are shown in Figure 23. Its failure characteristic is also similar to that of the Maxtor and WD disks. It has 31 bad sectors located on 4 tracks, 2 zones, and 27 of them are consecutive. Additionally, only 2 of the 4 surfaces are damaged. The same replication techniques proposed for Maxtor and WD disks should be applicable here.

Capacity	36.7 GB
Zones	8
Platters	2
Surfaces	4
RPM	10000
Interface	SCSI

Table 10. Quantum LCT10: Disk specification.

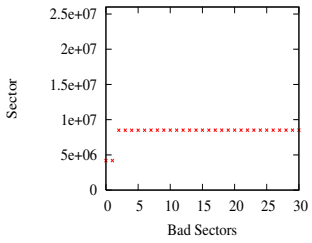

Figure 22. Quantum LCT10: Locations of bad sectors.

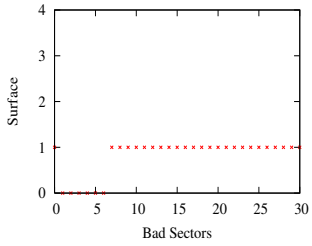

Figure 23. Quantum LCT10: Mapping of bad sectors to recording surfaces.

4.8. Quantum Fireball CX10.2A

Fireball CX10.2A is a slightly older disk than the LCT10. Its specification is shown in Table 11. 112 defective sectors (shown in Figure 24) were found and appeared to be randomly distributed on the disk—on all 3 recording surfaces and on 7 out of 15 zones. Only 21 sectors are co-located on the same track with any other defective sectors.

A closer examination of the defective sectors revealed something that we did not see in the other disks. As we can see from Figure 25, a large percentage of the defective sectors are mapped to surfaces #0 and #2. We then mapped the sectors to tracks and surfaces, which is shown in Table 12. This table shows that the failed sectors on each of these two surfaces are separated by exactly 6 tracks (or 2 cylinders). This can be caused by a head crash and the head has an odd shape, e.g., square wave.

Capacity	10.2 GB
Zones	15
Platters	2
Surfaces	3
RPM	5400
Interface	IDE

Table 11. Quantum CX10.2A: Disk specification.

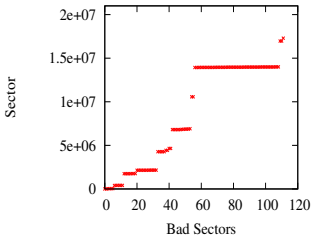

Figure 24. Quantum CX10.2A: Locations of bad sectors.

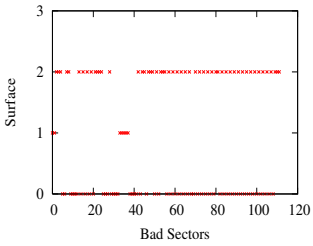

Figure 25. Quantum CX10.2A: Mapping of bad sectors to recording surfaces.

Sector	Track	Surface
13951235	30189	0
13953247	30194	2
13953571	30195	0
13955583	30200	2
13955907	30201	0
13957919	30206	2
13958243	30207	0
13960255	30212	2
13960579	30213	0
13962591	30218	2
13963319	30219	0
13965329	30224	2
13965330	30224	2
13965654	30225	0
13967666	30230	2
13967990	30231	0
13970002	30236	2
13970326	30237	0
13972338	30242	2
13972662	30243	0
13974674	30248	2
13975401	30249	0
13977413	30254	2
13977737	30255	0
13979749	30260	2

6 tracks
6 tracks
6 tracks
6 tracks
6 tracks

Table 12. Quantum CX10.2A: a sample mapping of defective sectors to their track and recording surface.

5. Related Work

As far as we know, there is no existing work that has studied partial disk failures in as much detail as we have done here. The emphasis of this paper is in the analysis of disk failures, and our proposed disk geometry extraction algorithm is only the means to achieve this goal. The proposed extraction algorithm is most related to those in [2, 5, 16] using microbenchmarks to infer various disk parameters. Work done by Worthington [17] and Schindler [13], on the other hand, can extract even more disk parameters and more quickly. However, their approach relies on interrogative SCSI commands, and therefore, cannot be used for IDE disks. Instead of trying to extract as much information about disks as possible, our algorithm only extracts those that are needed to do failure analysis. We specificly designed our probing method to be as quickly and efficient as possible in obtaining the needed parameters, so the extraction process can be performed either online (low overhead) or offline (fast).

Related to characterizing damages on disk recording surfaces, there is a large body of work [3, 8] in the area of tribology. They study how particles generated and accumulated within disk enclosure can cause damages to recording surfaces. Even though they characterized the extent of damage to recording surfaces, they have not studied the correlation between data fidelity and such physical damage.

Several researchers [4, 6, 7, 12] proposed how transparent data and file replication can be used to enhance disk's I/O performance and to prevent user mistakes from corruption the file system. Prabhakaran *et al.* [11] described how to properly handle partial disk faults in the file system and that failure modes other than "fail stop" [14] should be commonly expected in modern disks. They showed that commodity file systems are often poorly designed to handle such errors.

6. Conclusion

In this paper, we proposed a general approach and the necessary tools to analyze partially failed disks. From this study, using real disks, we found some interesting patterns in defective sectors' physical locations. We observed both common failure features among disks of the same manufacturer and also those that are common across multiple manufacturers. We also observed some failure characteristics that are unique to certain manufacturers.

Due to privacy concerns, getting failed disks was very difficult. From a limited number of failed disks that we have obtained from several sources, common failure features are apparent, but to draw a definite conclusion and for it to be of practical usage to file system designers, a much larger disk population is needed. It is our hope that users

and system administrators can use this tool and contribute disk failure data to a common database. We have specifically designed our probing tools to be low-overhead and fast, but to make the tools more usable for average users, we are making some more improvements—more transparent and platform independent.

References

[1] PC guide. *http://www.pcguide.com/ref/power/index.htm*.

[2] M. Aboutabl, A. Agrawala, and J. Decotignie. Temporally determinate disk access: An experimental approach. *Technical Report CS-TR-3752*, 1997.

[3] S. Chandra and B. Bhushan. Effect of particulate contamination on the friction and wear of a magnetic head–rigid disk interface. *Proceedings of the I Mech E Part J Journal Engineering Tribology*, 214(5), 2000.

[4] B. Cornell, P. Dinda, and F. Bustamante. Wayback: A user-level versioning file system for linux. *USENIX Annual Technical Conference*, 2004.

[5] Z. Dimitrijevic *et al.* Diskbench: User-level disk feature extraction tool. *Technical Report UCSB-TR-2004-18*, 2004.

[6] G. Ganger, M. Mckusick, C. Soules, and Y. Patt. Soft updates: A solution to the metadata update problem in file systems. *ACM Transactions on Computer Systems*, 18(2), 2000.

[7] H. Huang, W. Hung, and K. G. Shin. Fs2: Dynamic data replication in free disk space for improving disk performance and energy consumption. *Symposium on Operating System Principles (SOSP)*, pages 263–276, 2005.

[8] B. Liu, S. H. Soh, A. Chekanov, S. B. Hu, and T. S. Low. Particle build-up flying sliders and mechanism study of disk wear and head-disk interface failure in magnetic disk drives. *IEEE Transaction on Magnetics*, 32(5), 1996.

[9] P. Lyman and H. R. Varian. How much information? *http://www2.sims.berkeley.edu/research/projects/how-much-info-2003*.

[10] D. Patterson, G. Gibson, and R. Katz. A case for redundant arrays of inexpensive disks (raid). *Proceedings of ACM SIGMOD*, pages 109–116, 1988.

[11] V. Prabhakaran *et al.* IRON file systems. *Symposium on Operating System Principles (SOSP)*, 2005.

[12] D. Santry *et al.* Deciding when to forget in the elephant file system. *Symposium on Operating System Principles SOSP*, 1999.

[13] J. Schindler and G. Ganger. Automated disk drive characterization. *Technical Report CMU-CS-99-176*, 1999.

[14] F. B. Schneider. Implementing fault-tolerant services using the state machine approach. *ACM Computing Surveys*, 22(4), 1990.

[15] T. J. Schwarz, Q. Xin, E. L. Miller, D. D. Long, A. Hospodor, and S. Ng. Disk scrubbing in large archival storage systems. *IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2004.

[16] N. Talagala, R. Arpaci-Dusseau, and D. Patterson. Microbenchmark-based extraction of local and global disk characteristics. *Technical Report CSD-TR-99-1063*, 1999.

[17] B. L. Worthington *et al.* Online extraction of SCSI disk drive parameters. *Proceedings of the ACM SIGMETRICS*, pages 146–156, 1995.

[18] E. Zadok, J. Osborn, A. Shater, C. Wright, K.-K. Muniswamy-Reddy, and J. Nieh. Reducing storage management costs via informed user-based policies. *IEEE Conference on Mass Storage Systems and Technologies*, 2004.

0-7695-3025-7/07 $25.00 © 2007 IEEE