SMaRT: An Approach to Shingled Magnetic Recording Translation

Weiping He and David H.C. Du

Department of Computer Science, University of Minnesota, Twin Cities

Abstract

Shingled Magnetic Recording (SMR) is a new technique for increasing areal data density in hard drives. Drive-managed SMR (DM-SMR) drives employ a shingled translation layer to mask internal data management and support block interface to the host software. Two major challenges of designing an efficient shingled translation layer for DM-SMR drives are metadata overhead and garbage collection overhead.

In this paper we introduce SMaRT, an approach to **S**hingled **Ma**gnetic **R**ecording **T**ranslation which adapts its data management scheme as the drive utilization changes. SMaRT uses a hybrid update strategy which performs in-place update for the qualified tracks and out-of-place updates for the unqualified tracks. Background Garbage Collection (GC) operations and on-demand GC operations are used when the free space becomes too fragmented. SMaRT also has a specially crafted space allocation and track migration scheme that supports automatic cold data progression to minimize GC overhead in the long term.

We implement SMaRT and compare it with a regular Hard Disk Drive (HDD) and a simulated Seagate DM-SMR drive. The experiments with several block I/O traces demonstrate that SMaRT performs better than the Seagate drive and even provides comparable performance as regular HDDs when drive space usage is below a certain threshold.

1 Introduction

Perpendicular magnetic recording technique used by traditional HDDs is reaching its areal data density limit. SMR addresses this challenge by overlapping the neighboring tracks. Assuming the write head is two-track wide in an SMR drive, a write will now impact 2 tracks. That is, writing to a track may destroy the valid data in its adjacent track. Consequently data is preferred to be written onto the tracks in a sequential manner. However, random read is still supported by SMR. In the scope of this paper, for simplicity we assume the write head width is 2 tracks.

Tracks in SMR drives are usually grouped into logical units called "bands". Band size depends on specific designs. There are generally two types of SMR drives: the drive-managed SMR (DM-SMR) drives and the host-managed/host-aware SMR (HM-SMR/HA-SMR) drives. DM-SMR drives, such as the Seagate Archive HDD (8 TB) [4] currently available on the market, maintain a logical block addresses (LBAs) to physical block address (PBAs) mapping layer and therefore provide block interface to the host software such as file systems and databases. As a result, they can be used to replace the traditional HDDs without changes to the upper level applications. On the other hand, HM-SMR and HA-SMR drives are simply raw devices and rely on specially designed upper level applications to interact with the PBAs directly.

Depending on the update strategy, DM-SMR drives can further be classified into in-place update SMR (I-SMR) drives and out-of-place update SMR (O-SMR) drives. To perform an update operation to a previously written track in an I-SMR drive, data on the following tracks has to be safely read out first and then written back to their original positions after the data on the targeted track has been updated. To minimize this overhead, only a few tracks (4 or 5) per band are used to avoid long update propagation in some designs. There will be enough separation between any two adjacent bands called "safety gap" such that writing to the last track of each band will not destroy the valid data in the following band. Importantly, static LBA-to-PBA mappings are possible in I-SMR drives requiring no mapping tables and GC operations. However, a considerable percentage of drive space may be consumed for safety gaps due to small band size. For example, at least 20% of the total space is used as safety gaps if the band size is 4 tracks [10]. Generally, a bigger band size provides better space gain but worse

update performance.

O-SMR drives provide much more space gain by using larger bands or zones. Therefore only a negligible amount of space is used for safety gaps. To perform an update operation in O-SMR drives, the updated data will first be written to a new place and the old data will be invalidated. Those invalidated data must be reclaimed later by GC operations for reusing. A mapping table is required to keep track of these data movements. Therefore, challenges exist for designing efficient O-SMR drives which include the metadata overhead and the GC overhead.

In order for O-SMR drives to be adopted in the current storage systems and used for primary workloads (in addition to cold workloads), metadata overhead and GC overhead must be minimized. In this paper we propose a SMaRT scheme, a track-based shingled magnetic translation layer for DM-SMR drives that supports an address mapping at the block level. SMaRT is motivated by two unique properties of SMR. First, an unused/free track can serve as a safety gap for the preceding track to qualify for in-place updates. Second, different from an invalidated page in Solid State Drives, an invalidated track in SMR drives is essentially a free track and can be immediately reused as long as its next track is free too. Based on this property, SMaRT adopts a hybrid track update strategy which maintains a loop of track invalidating and reusing that minimizes the need of triggering GC operations. GC operations are therefore invoked only when the free SMR drive space becomes too fragmented.

Two major modules are designed in SMaRT in order to fully exploit these two properties. One is a track level mapping table and the other is a space management module. The former supports LBA-to-PBA mapping or block interface to the host software and the latter supports free track allocations and GC operations. During a GC operation, valid tracks are migrated to create bigger contiguous free space. Our design of the space management module also enables SMaRT to support an automatic cold data progression feature which can separate cold data from frequently updated or hot data over time to minimize GC overhead.

We implement SMaRT and compare it to a regular HDD and a simulated Seagate DM-SMR drive described in Skylight [6]. The experiments with several workloads demonstrate that SMaRT performs better than this Seagate DM-SMR drive and nearly as well as a regular HDD.

The remainder of the paper is organized as follows. Section 2 discusses the different layouts for I-SMR drives and O-SMR drives. Some related studies are introduced in Section 3. SMaRT is described in Section 4. Experiments and evaluations are presented in Section 5 and some conclusion is made in Section 6.

2 SMR Layout

SMR drives generally follow the geometry of regular HDDs except the tracks are overlapped. Similar to HDDs, each SMR drive may contain several platters. Physical data blocks are also addressed by Cylinder-Head-Sector (CHS). Since outer tracks are larger than inner tracks, the SMR drive space is divided into multiple zones. Tracks in the same zone have the same size. Each zone can be further organized into bands if needed. A small portion (about 1% to 3%) of the total space is usually used as unshingled random access zone (RAZ) or conventional zone for persistent metadata storage[7, 13, 14].

I-SMR drives and O-SMR drives organize and use the bulk shingled access zone (SAZ) differently. Drive-managed I-SMR drives usually organize the tracks into small bands for a good balance between space gain and performance as discussed and evaluated in [10]. Most existing work on O-SMR drives divide the shingled access zone into an E-region and an I-region. Sometimes multiple E-regions and I-regions may be used. E-region is essentially a persistent cache space organized as a circular log and used for buffering incoming writes, while I-region is used for permanent data storage and organized into big bands. Obviously, writes to E-region and I-region have to be done in a sequential manner and GC operations are required for both regions. The E-region size is suggested to be no more than 3% [7, 8, 9, 14].

3 Related Work

There are a few studies that have been done for I-SMR drives. Shingled file system [13] is a host-managed design for I-SMR where the file system directly works on SMR drive PBAs. The SMR drive space is organized into bands of 64 MB. Files are written sequentially from head to tail in a selected band. He *et al.* proposed several static address mapping schemes for drive-managed I-SMRs [10]. The I-SMR drive space is organized into small bands of four tracks. By changing the order of utilizing the tracks, the new address mapping schemes can significantly reduce write amplification overhead compared to the traditional mapping scheme. However, a non-ignorable percentage of total capacity (about 20%) has to be used as safety gaps between neighbouring bands in order to achieve desired performance.

Several studies have also been done for drive-managed O-SMR drives. For example, Cassuto *et al.* proposed two indirection systems in [8]. Both systems use two types of data regions, one for caching incoming write requests and the other for permanent data storage. They proposed an S-block concept in their second scheme. S-blocks have the same size and each S-block consists of a

pre-defined number of sequential regular blocks/sectors such as 2000 blocks as used in [8]. GC operations have to be performed in both data regions in an on-demand way. Hall *et al.* proposed a background GC algorithm [9] to refresh the tracks in the I-region while data is continuously written into the E-region buffer. The tracks in the I-region have to be sequentially refreshed at a very fast rate in order to ensure enough space in the E-region, which is expensive and creates performance and power consumption issues.

Host-managed O-SMR is another new design trend. Jin *et al.* proposed the HiSMRfs [11] which is a host-managed solution. HiSMRfs pairs some amount of SSD with the SMR drive so that file metadata (hot data) can be stored in the SSD while file data (cold data) can be stored in the SMR drive. HiSMRfs uses file-based or band-based GC operations to reclaim the invalid space created by file deletions and file updates. However, the details of the GC operations are not discussed. Caveat-Scriptor [12] is another host-managed design which protects valid data with a Drive Isolation Distance (DID) and a Drive Prefix Isolation Distance (DPID). It is implemented as SMRfs which is a simple FUSE-based file system. Caveat-Scriptor supports a Free Cleaning scheme to delay and reduce on-demand GC operations. Different from HiSMRfs and Caveat-Scriptor, SMRDB [17] is a host-managed design for key-value store that is file system free. SMRDB defines a set of data access operations including GET/PUT/DELETE/SCAN to comply with the successful KV data access model used in recent cloud storage systems.

The disk drive industry has introduced several openly available SMR drives on the market including the Seagate Archive HDD (8 TB) [4] and the Western Digital Ultrastar Archive Ha10 (10 TB) [5]. These drives are specially designed for cold workloads such as archive systems and backup systems. Aghayev and Desnoyers conducted a series of microbenchmarks and video recording through a drilled hole on the drive shell to reverse engineer Seagate SMR drives [6]. Their observations revealed the presence of an on-disk persistent cache with a lazy write back policy and they further studied other aspects including the cache size and band size. We use these information to implement a simulated Seagate DM-SMR drive in our experiments. Recently, Wu *et al.* conducted evaluations on special features of real HA-SMR sample drives such as open zone issue and media cache cleaning efficiency. They also proposed a host-controlled indirection buffer to improve I/O performance [18].

4 SMaRT Scheme

In this section, we describe the proposed SMaRT scheme and discuss its designs and implementations. There are two major function modules in SMaRT: a track-based mapping table (Section 4.1) that supports LBA-to-PBA mapping and a space management module (Section 4.2) that manages free track allocations and GCs.

4.1 Track Level Mapping Table

The first main function module of SMaRT is a LBA-to-PBA mapping table at a track level which enables SMR drives to communicate with the host software using the block interface. Therefore SMR drives can be used in existing storage systems in a drop-in manner. The mapping table is updated when: 1) A used track is updated to a new location, 2) Used tracks are migrated during GCs or 3) Tracks are allocated for new data.

Given an LBA, SMaRT will first calculate its corresponding logical track number (LTN) and its offset inside this track based on the number of zones and the track size in each zone. It then looks up the mapping table and translates LTN into physical track number (PTN). The final PBA can be easily computed with PTN and the offset inside the physical track.

Assuming an average track size of 1 MB, an 8 TB SMR drive requires at most a 64 MB track level mapping table (assuming 4 bytes for each LTN and PTN). The mapping table is initially empty and gradually grows as the space utilization increases. Therefore, the metadata overhead of the track level mapping table is reasonably low considering the fact that the standard DRAM size for large capacity HDDs/SMRs on the market today is 128 MB [4] or 256 MB [5].

As a comparison, the Seagate DM-SMR drive described in [6] uses a persistent cache (E-region). When adopting a block-level mapping scheme, an E-region of size 80 GB (= 1% of 8 TB) yields a 160 MB mapping table based on 4 KB block size or 1280 MB based on 512 B block size. Alternatively, extent mapping [6] can be used to reduce the mapping table size which however provides less effective persistent cache size to the host writes.

4.1.1 Hybrid Track Update

SMaRT uses a hybrid track update strategy that updates qualified tracks in place and updates the unqualified tracks out of place. Track update has been proven to be beneficial and affordable because it creates a track invalidating and reusing loop. When a track is invalidated, it actually becomes a free track and can be reused as long as its next track is free or as soon as its next track becomes free. As a result, track updates in SMaRT continuously invalidate tracks and turn them into free tracks without triggering explicit on-demand GC operations. This greatly reduces the frequency of invoking

GCs which compensates for the cost of the track update operations. Explicit on-demand GC operations are only invoked when the free SMR space becomes too fragmented as discussed in Section 4.2.4.

4.2 Space Management

The second main component of SMaRT is a space management module which is responsible for free track allocation and garbage collections. Space management is done for each zone separately.

Free track allocation means that when a track is updated, SMaRT has to choose a new track position from the available usable free tracks. The updated track will be written to the new position and the old track is invalidated/freed.

As described previously, usable free tracks and unusable free tracks coexist. Garbage collection is essentially a free space consolidation that can turn unusable free tracks into usable free tracks by migrating the used tracks.

We now introduce a concept of "space element" and a "fragmentation ratio" before describing the details of SMaRT.

4.2.1 Space Elements

There are two types of tracks in an SMR drive: the used (or valid) tracks and the free tracks. When a used track is invalidated, it becomes a free track but it is not considered as a usable free track if its following track is not free. However, a free but unusable track can at least serve as a safety gap and allow its preceding track to be updated in place.

All the used tracks constitute the **used space** and all the free tracks constitute the **free space**. We call a group of consecutive used tracks or free tracks a *space element* which is used to describe the current track usage. For the example of Figure 1, we say the used space includes elements {0, 1, 2, 3}, {6}, {10, 11, 12}, {14, 15, 16, 17} and {20, 21} while the free space includes elements {4, 5}, {7,8,9}, {13}, {18, 19} and {22, 23}. The **size** of a particular space element is defined as the number of tracks in it. The last track in each free space element is not usable and can not be written because writing to this last track will destroy the valid data on the following track which is a used track. Particularly, a free space element of size 1 contains no usable free track such as element {13}. The number of elements and their sizes continuously change as incoming requests are processed. A free track that is previously unusable can become usable later as soon as its following track becomes free too. Accordingly the last track in a used space element can be updated in place because its next track is a free track.

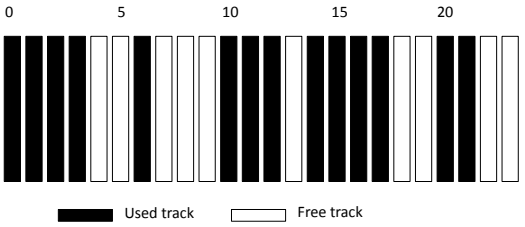

Figure 1: SMR Usage State

4.2.2 Fragmentation Ratio

We define the free space fragmentation ratio to help decide when to invoke on-demand GCs in each zone. Assuming the total number of **free tracks** in a selected zone is F and the total number of **free space elements** is N, the free space *fragmentation ratio* (R) for this zone can be computed according to Equation 1. In fact, the fragmentation ratio represents the percentage of usable free tracks in all the free tracks. Fragmentation ratio of 0 means the free space is too fragmented. In fact, 0 means all free space elements are of size 1 and thus no track can be used.

$$R = \frac{F - N}{F}, where\ 1 \leq N \leq F \tag{1}$$

We conduct a series of permutation tests to study the impacts of the fragmentation ratio threshold (R) on SMaRT performance which show that neither a small R threshold nor a large R threshold produces good performance. A small R makes SMaRT adopt a lazy garbage collection strategy. On-demand GC operations are only invoked when the free space is extremely fragmented which requires more victim elements to be migrated in a single GC operation and causes I/O burstiness. A big R ratio is not suggested either since frequent unnecessary GCs will be invoked even though the free space is not fragmented. A larger ratio also means a smaller N and thus a smaller number of tracks that support in-place updates. We use 0.5 in our experiments to allow SMaRT to maintain relatively big contiguous free space and trigger a GC only if necessary.

4.2.3 Space Allocation

SMaRT always designates the largest free space element as an "allocation pool" and maintains a dedicated track pointer called "*write cursor*" that initially points to the first track of the allocation pool, as shown in Figure 2a. The free tracks in this allocation pool are allocated to accommodate updated tracks as well as new write data in a sequential manner in the shingling direction. A modified track is always written to the free track pointed by the

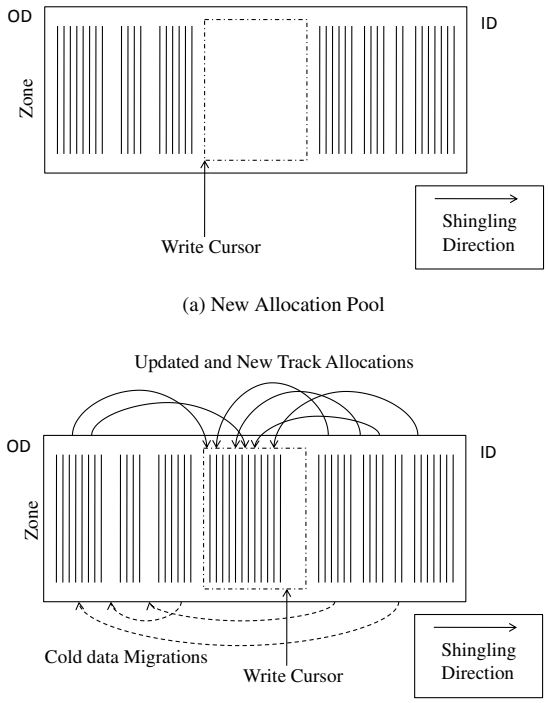

Figure 2: The Process of SMaRT

write cursor which will be incremented accordingly. After the write cursor reaches the end of the allocation pool, SMaRT will select the latest largest free space element as the new allocation pool and update the write cursor to its first track.

Specially, SMaRT defines data that is recently updated as hot. New write (first write) data and data not updated for some time will be treated as cold. For example, the least recently updated track is definitely cold and the most recently updated track is hot. SMaRT will allocate an extra guarding track for a hot data track when the drive utilization is lower than a certain threshold (such as 50%) because there are a sufficient amount of free tracks to be allocated as safety tracks. Therefore these hot data tracks now support in-place updates which reduces unnecessary out-of-place updates. This feature is disabled once the space utilization goes over a threshold and will be re-enabled when space utilization falls below the threshold. The transition is fully transparent.

4.2.4 Space Consolidation

A garbage collection in SMaRT is essentially a free space consolidation. There are two types of GCs in SMaRT: background GCs and on-demand GCs. Background GC operations are only performed during the drive idle times. When a time period between two writes is longer than a threshold value, it is considered as an idle time [6]. Background GC operations keep running until 1) space fragmentation ratio surpasses its threshold or 2) idle time ends. On the other hand, the fragmentation ratio is checked upon each incoming write request. If it is equal to or smaller than the threshold value, an on-demand GC operation will be invoked to migrate used tracks and combine the small free space elements into bigger elements. This will improve I/O performance for big writes and updates, as well as increase usable free space. An on-demand GC usually immediately stops after one element has been moved so as to minimize the overhead of a single GC operation. This minimizes the performance interference on serving the incoming requests. The fragmentation ratio sometimes remains below its threshold after the current GC operation. The next GC operation in this zone will continue to improve the fragmentation. Moving multiple elements happens only when there is not enough usable free tracks to accommodate the updated track(s) or new tracks.

To perform a GC operation, SMaRT searches for a victim element starting from the leftmost used space element of this zone and proceeding in the shingling direction. A victim element has a size of smaller than a threshold W. W is initialized to be a small value based on the current SMR drive space usage U. It can be calculated according to Equation 2. In fact, W practically represents the used space to free space ratio. The theory behind this equation is that the average used space element size is bigger when the SMR space utilization is higher. For example, W will be initially set to 2 if the current SMR drive usage is 60% or 9 if the usage is 90%. If no element is found to be smaller than W, W will be doubled and SMaRT will redo a search based on the new W. This will be repeated until a satisfying element is found.

$$W = \frac{U}{1-U}, where\ 0 < U < 1 \tag{2}$$

SMaRT always tries to append a victim element to the leftmost free space element that fits the victim element[1]. If no free space to the left can accommodate the victim element, SMaRT simply shifts it left (against shingling direction) and appends it to its left used space element neighbour. Besides, if the victim element resides in the allocation pool, it will also be appended to the left neighbour because this victim element contains recently written/updated tracks and thus is deemed as possibly hot which should not be appended to cold data.

In Figure 1, R is 0.5 by Equation 1 and W is 2 by Equation 2. Assuming an on-demand GC is triggered, SMaRT will select the used space element $\{6\}$ as the victim and

[1]The used space element containing track 0 is a special case.

append it to element {0, 1, 2, 3}. Consequently, the free space elements {4, 5} and {7, 8, 9} will be consolidated into a single bigger element {5, 6, 7, 8, 9}. The resulting fragmentation ratio R is 0.6. SMaRT will detect upon the next request that R is above the threshold and thus it will not invoke another GC operation.

4.2.5 Cold Data Progression

The track-based mapping and data migrations provide a good opportunity of automatic cold data progression. This is achieved as part of the free track allocation and GC operations, requiring no dedicated hot/cold data identification algorithms. The cold data progression in SMaRT is illustrated in Figure 2b. The allocation pool accumulates the recently updated tracks or hot tracks. Cold data gets migrated against the shingling direction to the left by GC operations. Eventually the cold data will mostly stay at the left side of the zones and hot data gets updated and pushed to the right side of the zones which reduces unnecessary cold data movements during GC operations.

The downside of this is that cold data (least recently updated data) will eventually reside on the outer tracks which possibly wastes the sequential I/O performance of the outer tracks. We provide one argument and one solution to this concern. We argue that cold data can be frequently read by our definition. Least recently updated data can be most frequently read data and/or most recently read data.

To directly address this concern, we can reverse the shingling direction by shingling from ID (Inner Diameter) to OD (Outer Diameter) so that writes start with inner tracks and move toward outer tracks. This way, cold data will be gradually moved to the inner tracks. Note that we assume the normal shingling direction in our experiments.

4.3 I/O Processing Summary

The overall I/O procedure for read requests is straightforward. SMaRT simply reads the data after translating the LBAs into PBAs. Multiple reads will be issued if read is fragmented.

On receiving a write request, SMaRT first checks the fragmentation ratio to decide if a GC operation should be invoked. After a necessary GC operation completes, SMaRT checks whether the write request operates on existing data by consulting the mapping table. For new data, SMaRT will allocate free tracks and add corresponding new mapping entries. For existing data, SMaRT checks whether a track supports in-place update. If not, SMaRT allocates new tracks and updates the existing mapping entries.

During the workload, once an idle time is identified and meanwhile the free space fragmentation ratio is below the threshold, background GCs will be launched.

4.4 SMaRT for Cold Write Workload

SMaRT performs extremely like a regular HDD for cold workloads such as backup, archive and RAID rebuild workloads. Data will be written sequentially into the allocation pool without extra guarding safety tracks because new write data is treated as cold. Data can still be updated based on the hybrid update strategy if ever needed. No changes to the space management scheme are needed.

4.5 Reliability

The mapping table of SMaRT is periodically synchronized to a persistent mapping table copy in the random access zone on disk. However, there is still a risk that a power failure occurs before the latest mapping table updates are pushed to the disk. This is a common reliability challenge to flash translation layers (FTLs) and shingled translation layers (STLs). Here we sketch a low-cost scheme for SMaRT inspired by a *Backpointer-Assisted Lazy Indexing*[15].

Since SMaRT always writes new tracks and updated tracks to the allocation pool, SMaRT can simply trigger a synchronization whenever an allocation pool is fully consumed and flush the mapping table updates to the disk. Upon flush completion, SMaRT records the timestamp and picks a new allocation pool. Tracks updated after the latest synchronization are ensured to reside in the latest allocation pool only. When writing a new track or updating an existing track to a new position, a backpointer to the logical track number (LTN) along with the current timestamp will be stored together with the track such that each physical track internally has a PTN-to-LTN reverse mapping entry. We assume there will be some tiny spare space associated with each physical track that can be used to store the LTN and the timestamp. Otherwise, we can simply reserve a sector or block in each physical track to be used for storing these extra information.

To recover from a power failure, only the latest allocation pool needs to be scanned instead of the whole disk. The synchronization interval can also be the time to consume a list of most recent allocation pools. In this case, the manufacturing cost of SMR drives is minimal because no extra media is introduced. To perform a recover, SMaRT scans the latest allocation pool and identifies tracks with timestamps newer than that of the latest synchronization. SMaRT then reads their associated LTNs and PTNs to construct the corresponding LTN-to-PTN mapping entries which will be merged to the map-

ping table copy on disk so that the latest LTN-to-PTN table will be restored.

Alternatively, Non-Volatile Memory and flash media can be used to persist the mapping table when the cost of the former and the durability of the latter become acceptable.

5 Evaluations

In this section, we evaluate the performance of SMaRT and make comparisons with other schemes.

5.1 Competing Schemes

We compare SMaRT to a regular HDD and a simulated Seagate DM-SMR drive. Since our objective is to design SMR drives that can perform well under primary workloads in existing storage systems, we choose the regular HDD as the baseline for comparison. We are hoping the performance of the new design can be close to that of HDDs.

We also implement a Seagate DM-SMR drive based on the configurations and schemes explored and described in Skylight [6]. We denote this SMR drive as **Skylight** which is configured with a single 2 GB persistent cache at OD, 40 MB band size, static address mapping for bands and aggressive GC operation with a 1242ms triggering idle window. Incoming writes go to the cache first. A Skylight background GC would be triggered if there is a 1242ms idle time window in the workload since the last request. It keeps running until the next request comes in. A GC operation scans from the tail of the circular-log structured persistent cache, reads all the sibling blocks that belongs to the same band as the tail block and read-modify-writes the band. In our experiments, we find that on-demand GCs are also needed when there are not enough idle times in the workloads. An on-demand GC is triggered when the persistent cache space is 80% full.

5.2 Implementations

Due to the needs of defining zone sizes and controlling the starting/ending PBAs of the zones, we have to adopt simulation instead of using real hard disk drives. We implement these schemes on top of Disksim [2] to simulate an SMR drive based on the parameters of a Seagate Cheetah disk drive [1]. This is the newest and largest validated disk model that is available to Disksim. It has a capacity of 146 GB (based on 512 B sector size) [2]. We divide the total capacity into 3 parts: one 2 GB random access zone, one 2 GB perstistent cache (i.e., E-region) and the rest of the 142 GB for persistent storage space (i.e., I-region). The random access zone and the E-region will not be allocated if they are not used in a specific scheme. Particularly, both the random access zone and the E-region are not used in HDD. And the E-region is not used in SMaRT. On receiving disk I/O requests (in the form of LBAs), these schemes will translate the block addresses into PBAs based on their own implementation logic. The translated requests (in the form of PBAs) are then passed to the Disksim for processing.

[2]The drive capacity is about 1.1 TB based on 4 KB sector size.

Table 1: Trace Statistics

Trace	MAX LBA	factor	Req. Size	Write%
mds_0	71,127,264	5	18	88.11%
proj_0	34,057,712	10	76	87.52%
stg_0	22,680,944	12	23	84.81%
rsrch_0	35,423,624	10	17	90.67%

5.3 Trace Information

Four write intensive MSR traces [3, 16] are used in our experiments since other read intensive traces may not trigger media cache cleaning. The characteristics of the MSR traces are shown in Table 1 which include the maximum LBA, the average request size (R.S.) in blocks and the write ratio. We scale up the LBAs in the four traces to make sure the whole 142GB space is covered. Based on the maximum LBA, a different scale up factor is used for each trace.

5.4 Experiment Design

We test the schemes at different drive space utilizations including 30%, 60% and 90% which allow us to understand the impact of space utilizations on the drive performance. The drives will be pre-populated with data according to the utilization which is done by manipulating the metadata or mapping table in each scheme. Oversized LBAs in the traces will be trimmed with modular operation to fit in the accessed LBA range.

5.5 Performance Comparison

We use *response time*, *Write Amplification*, *Read Fragmentation* and *GC Overhead* as the main performance comparison metrics.

5.5.1 Response Time

The response time for each request is calculated by subtracting the queueing time stamp from the request completion time stamp. The overall average response times

for different schemes under different workloads at different drive space utilizations are shown in Figures 3, 4 and 5. X-axis is the average response time and Y-axis is the corresponding Cumulative Distribution Function (CDF).

The results show that in general HDD performs best for all utilizations. SMaRT has better performance than that of Skylight for the 30% and 60% utilizations and the two schemes provide a comparable performance for the 90% utilization. The Skylight only shortly crosses over the HDD and SMaRT in some of the low response time ranges while lags behind for the majority of the response times. The cross-overs are mainly contributed by more physically sequential writes due to the using of persistent cache.

5.5.2 Read Fragmentation

SMaRT suffers read fragmentation because of track movements. A read request can span over multiple logically consecutive but physically scattered tracks. Skylight incurs read fragmentation because of persistent cache. Some blocks in a read request may exist and scatter inside the persistent cache while the rest blocks still reside in the bands. As a result, a single read request can be fragmented into several smaller requests in both schemes, although the total number of blocks accessed remains the same.

We first show the percentages of fragmented reads in both schemes in Figure 6 and then compare the read fragmentation ratio in Figures 7, 8 and 9. We define the read fragmentation ratio as the number of "smaller read requests" produced by a single fragmented read request. Theoretically, the ratio is upper-bounded by the number of tracks accessed by the read request for SMaRT while bounded by the number of blocks in the request for Skylight. This is also the reason why fragmented read percentages in SMaRT are less affected by the space utilizations as seen in the figure.

As expected, the result shows that SMaRT has higher fragmented read percentages (Figure 6) with lower fragmentation ratios (Figures 7, 8 and 9). In general, more than 95% of the fragmented reads have a fragmentation ratio of 3 for SMaRT.

5.5.3 Write Amplification

We define *amplified write percentage* as the percentage of the write requests that trigger on-demand GCs outside the idle times. The result is shown in Figure 10.

Note that the amplified write percentages for 30% are all zero for SMaRT. No GC operation is incurred because the space allocation scheme in SMaRT will allocate guarding safety tracks when space utilization is less than 50% which suffices to maintain a good free fragmentation ratio so as to avoid triggering on-demand GCs.

Table 2: GC Statistics

Traces	SMaRT		Skylight	
30%	GC_F	GC_B	GC_F	GC_B
mds_0	0	0	1588	118417
proj_0	0	0	1861	97314
stg_0	0	0	1823	72229
rsrch_0	0	0	2003	108665
60%	GC_F	GC_B	GC_F	GC_B
mds_0	2045	603	1840	118417
proj_0	1560	536	2278	97314
stg_0	2583	1203	2191	72229
rsrch_0	3544	789	3099	108665
90%	GC_F	GC_B	GC_F	GC_B
mds_0	18367	5226	1866	118417
proj_0	13040	4595	2646	97314
stg_0	50131	23029	2834	72229
rsrch_0	35416	9722	3921	108665

However, as space utilization increases to 60% and 90%, up to 6% of the write requests will trigger on-demand GCs compared to 0.5% for Skylight. These percentage numbers are low mainly because of the contributions of background GCs.

5.5.4 GC Overhead

As described previously, there are two types of GC operations in both schemes: on-demand GCs (GC_F) and background GCs (GC_B). Table 2 shows the average numbers of on-demand GCs and background GCs incurred per 1 millions write requests. The table covers different traces and space utilizations.

In general, these numbers increase as the space utilization climbs. SMaRT triggers more on-demand GCs and Skylight triggers more background GCs. The fact that SMaRT relies more on on-demand GCs makes it theoretically more suitable for workloads with less idle times.

Specially, the numbers of both the on-demand GCs and the background GCs are 0s for the 30% utilization. SMaRT also uses a 1242ms idle time window as the first background GC triggering condition and the free space fragmentation ratio as the second condition. A background GC would be launched when both conditions are met and stops when the free space fragmentation ratio goes above the threshold or when the next write arrives. As a result, the 0s are simply because the free space fragmentation ratio stays high enough that no GC is needed.

Additionally, the numbers of background GCs for Skylight stay the same across different utilizations because Skylight starts background GCs when a 1242ms idle window is detected and stops when the next request arrives which is not affected by the utilization changes.

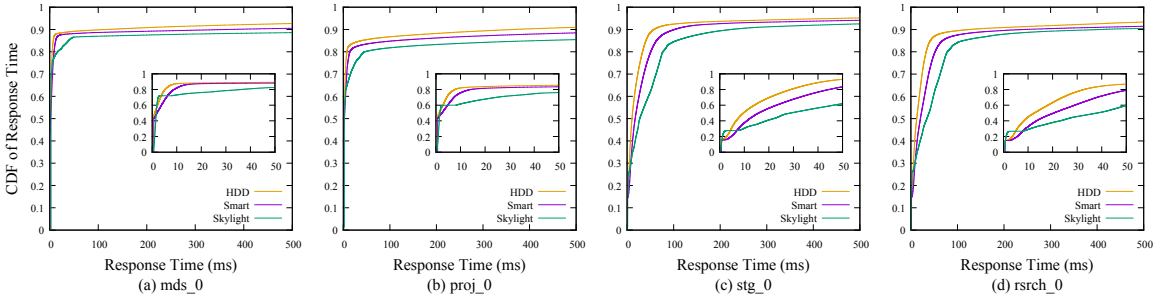

Figure 3: CDF of Response Time at 30% Utilization

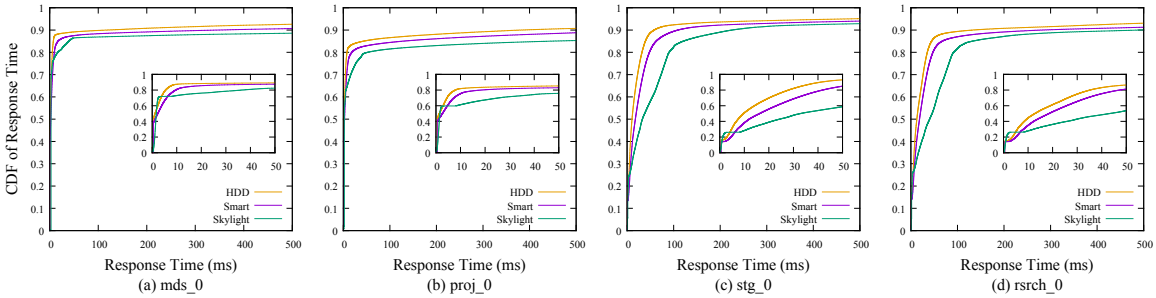

Figure 4: CDF of Response Time at 60% Utilization

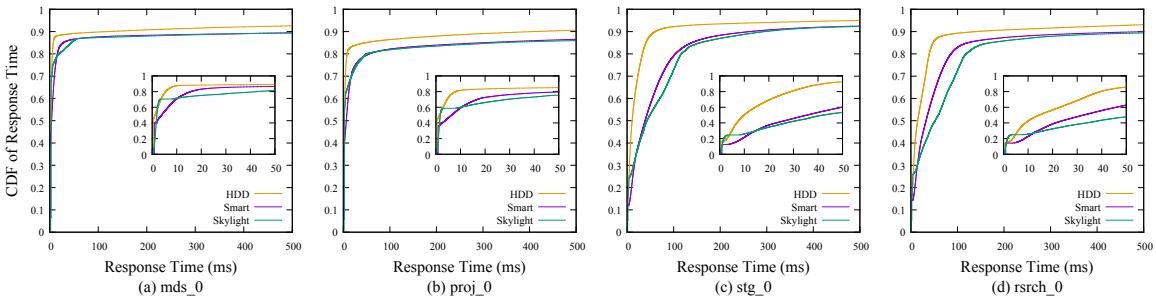

Figure 5: CDF of Response Time at 90% Utilization

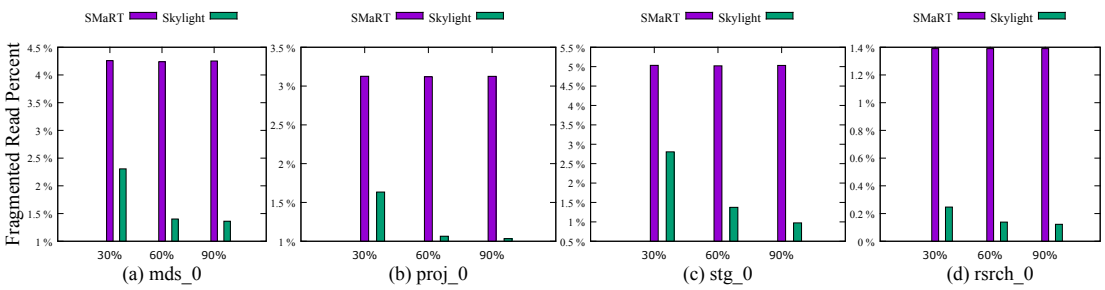

Figure 6: Fragmented Read Percent

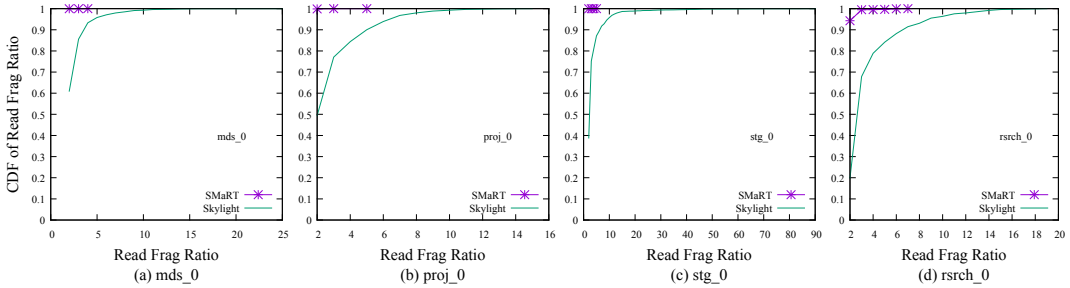

Figure 7: CDF of Read Frag Ratio at 30% Utilization

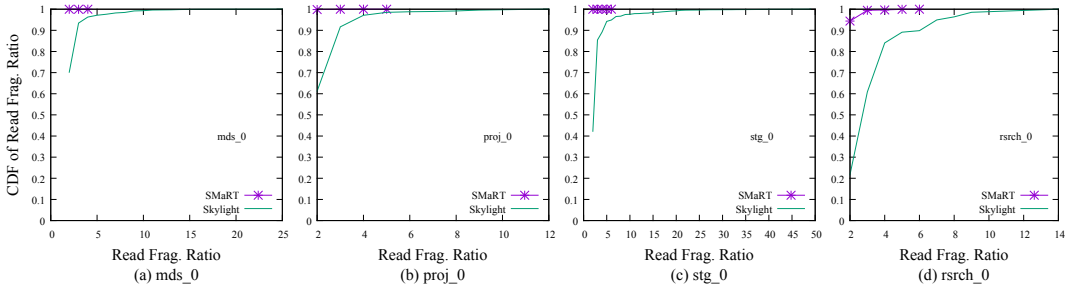

Figure 8: CDF of Read Frag Ratio at 60% Utilization

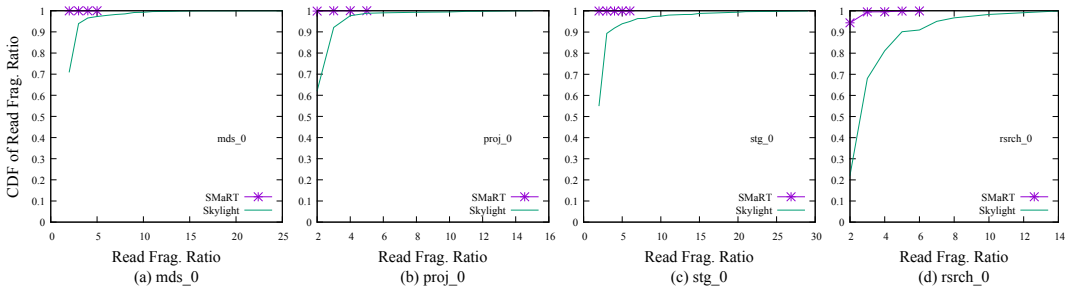

Figure 9: CDF of Read Frag Ratio at 90% Utilization

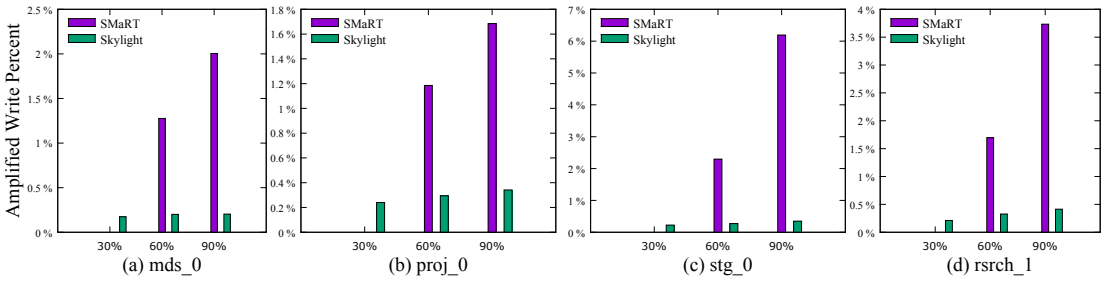

Figure 10: Amplified Write Percent

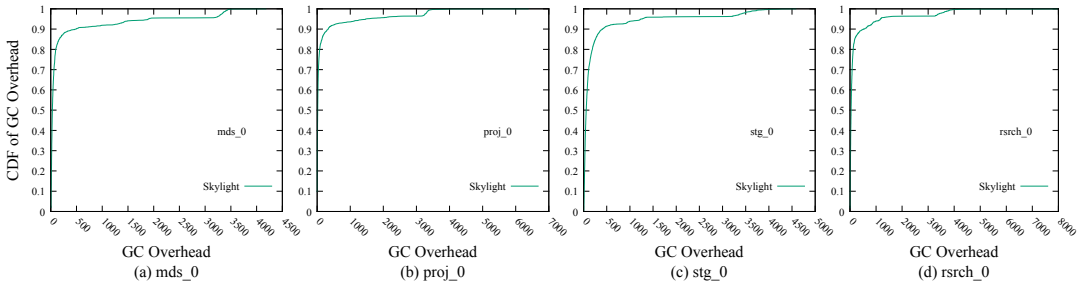

Figure 11: CDF of GC Overhead at 30% Utilization

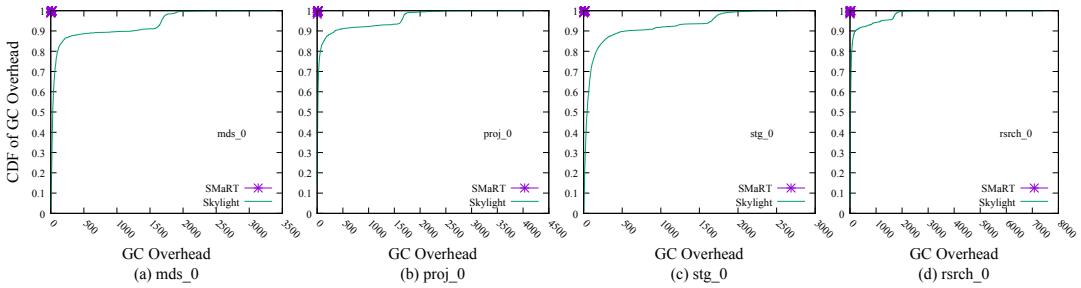

Figure 12: CDF of GC Overhead at 60% Utilization

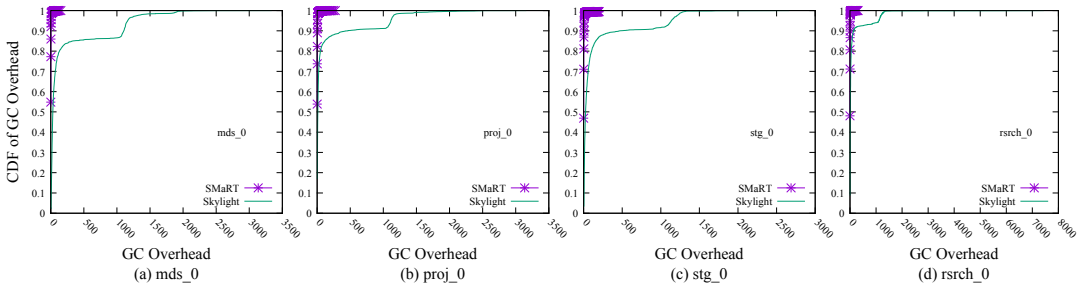

Figure 13: CDF of GC Overhead at 90% Utilization

Table 3: Scheme Comparison Summary

Schemes	E-region?	Performance	Cold Workload Friendly	Metadata Overhead	Space Gain
HDD	no	very good	yes	no	100%
SMaRT	no	good	yes	low	>95%
Skylight	yes	fair	no	high	>95%

Since we have shown the GC counting, we now discuss the on-demand GC overhead shown in Figures 11, 12 and 13. GC overhead (X-axis) is defined differently for SMaRT and Skylight because of the different natures of their GCs, although the two schemes share the X-axis. It is defined as the number of tracks migrated during a GC operation for SMaRT while defined as the total number of I/O requests needed for a single GC operation for Skylight. So there is no direct performance comparison here. Instead, this serves as helping understand the overhead origination in each GC scheme. About 50% of SMaRT's on-demand GCs move a single track and about 99% of the GCs move a single-digit number of tracks. On the other hand, Skylight's on-demand GCs have a wide overhead spectrum. However, about 80% of its on-demand GCs consists of less than 100 I/O requests.

5.5.5 Comparison Summary

We now summarize the comparison results in Table 3. The schemes are compared according to different metrics including performance, cold write workload suitability and metadata overhead. SMaRT is the preferred design which achieves a good balance among the listed metrics. It produces better performance than that of Skylight for low-range and mid-range space utilizations. Besides, SMaRT is more friendly to cold workloads because it does not stage incoming data in a persistent cache and destage later as Skylight does. SMaRT also has a lower metadata overhead because of not using persistent cache.

6 Conclusion and Future Work

In this paper we propose an efficient SMaRT scheme for drive-managed SMR drives by exploiting two unique properties in the SMR drives. SMaRT performs copy-on-write updates only when in-place updates are impossible. The track level mapping, when combined with a novel space management scheme can automatically filter the cold data to minimize data migration overhead for GC operations. The Experiments with real world workloads demonstrate that SMaRT can perform as well as regular HDDs under cold write workloads and even under primary workloads when space usage is in the lower and middle ranges.

In the future, we plan on designing a request scheduling algorithm that bundles multiple write requests as a single request if they belong to the same track or adjacent tracks. This would further reduce the write amplification overhead and improve write performance especially for high disk utilization situations. We also plan to extract parameters from the latest SMR drives and validate the resulting drive model for further simulation purposes. We plan to also investigate the performance of SMaRT when the write head width is more than 2 tracks.

Acknowledgement

We thank the anonymous reviewers and our shepherd Erik Riedel for their insightful comments on earlier drafts of the work. This work was partially supported by NSF awards 130523, 1439622, and 1525617.

References

[1] Database of Validated Disk Parameters. `http://www.pdl.cmu.edu/DiskSim/diskspecs.shtml`.

[2] DiskSim. `http://www.pdl.cmu.edu/DiskSim/`.

[3] MSR Cambridge Block I /O Traces. `http://iotta.snia.org/traces/388`.

[4] Seagate Archive HDD. `http://www.seagate.com/products/enterprise-servers-storage/nearline-storage/archive-hdd/`.

[5] WD UltraStar Ha10. `http://www.hgst.com/products/hard-drives/ultrastar-archive-ha10`.

[6] A. Aghayev and P. Desnoyers. Skylight—a window on shingled disk operation. In *13th USENIX Conference on File and Storage Technologies (FAST 15)*, pages 135–149, Santa Clara, CA, Feb. 2015. USENIX Association.

[7] A. Amer, D. D. Long, E. L. Miller, J.-F. Paris, and S. Schwarz. Design issues for a shingled write disk system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–12. IEEE, 2010.

[8] Y. Cassuto, M. A. Sanvido, C. Guyot, D. R. Hall, and Z. Z. Bandic. Indirection systems for shingled-recording disk drives. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–14. IEEE, 2010.

[9] D. Hall, J. H. Marcos, and J. D. Coker. Data handling algorithms for autonomous shingled magnetic recording hdds. *Magnetics, IEEE Transactions on*, 48(5):1777–1781, 2012.

[10] W. He and D. H. Du. Novel address mappings for shingled write disks. In *6th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 14)*. USENIX Association.

[11] C. Jin, W.-Y. Xi, Z.-Y. Ching, F. Huo, and C.-T. Lim. Hismrfs: A high performance file system for shingled storage array. In *Mass Storage Systems and Technologies (MSST), 2014 30th Symposium on*, pages 1–6. IEEE, 2014.

[12] S. Kadekodi, S. Pimpale, and G. A. Gibson. Caveat-scriptor: write anywhere shingled disks. In *Proceedings of the 7th USENIX Conference on Hot Topics in Storage and File Systems*, pages 16–16. USENIX Association, 2015.

[13] D. Le Moal, Z. Bandic, and C. Guyot. Shingled file system host-side management of shingled magnetic recording disks. In *Consumer Electronics (ICCE), 2012 IEEE International Conference on*, pages 425–426. IEEE, 2012.

[14] C.-I. Lin, D. Park, W. He, and D. H. Du. H-swd: Incorporating hot data identification into shingled write disks. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2012 IEEE 20th International Symposium on*, pages 321–330. IEEE, 2012.

[15] Y. Lu, J. Shu, W. Zheng, et al. Extending the lifetime of flash-based storage through reducing write amplification from file systems. In *FAST*, pages 257–270, 2013.

[16] D. Narayanan, A. Donnelly, and A. Rowstron. Write off-loading: Practical power management for enterprise storage. *ACM Transactions on Storage (TOS)*, 4(3):10, 2008.

[17] R. Pitchumani, J. Hughes, and E. L. Miller. Smrdb: key-value data store for shingled magnetic recording disks. In *Proceedings of the 8th ACM International Systems and Storage Conference*, page 18. ACM, 2015.

[18] F. Wu, M.-C. Yang, Z. Fan, B. Zhang, X. Ge, and D. H. Du. Evaluating host aware smr drives. In *8th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 16)*. USENIX Association, 2016.