Evaluating Host Aware SMR Drives

Fenggang Wu Ming-Chang Yang[†] Ziqi Fan Baoquan Zhang
Xiongzi Ge David H.C. Du
University of Minnesota, Twin Cities [†]*National Taiwan University*

Abstract

Shingled Magnetic Recording (SMR) technology increases the areal density of hard disk drives. Among the three types of SMR drives on the market today, Host Aware SMR (HA-SMR) drives look the most promising. In this paper, we carry out evaluation to understand the performance of HA-SMR drives with the objective of building large-scale storage systems using this type of drive. We focus on evaluating the special features of HA-SMR drives, such as the open zone issue and media cache cleaning efficiency. Based on our observations we propose a novel host-controlled indirection buffer to enhance the drive's I/O performance. Finally, we present a case study of the open zone issue to show the potential of this host-controlled indirection buffer for HA-SMR drives.

1 Introduction

In order to further increase the areal density of hard disk drives, Shingled Magnetic Recording (SMR) technology places the data tracks in a shingled (sequentially overlapped) fashion, while still ensuring the data can be read from the uncovered portion of the tracks [1].

There are three types of SMR drives on the market today, i.e. *drive managed*, *host managed*, and *host aware* SMR drives [2]. Drive Managed SMR (DM-SMR) drives redirect updating I/O requests (or non-sequential writes) to *media cache* (or *persistent cache*, one part of the disk) and later migrate those redirected data blocks back to their original places via the *cleaning* process. Such media cache operations, i.e. redirection and cleaning, are transparent to the host [3][4]. Host Managed SMR (HM-SMR) drives, in contrast, expose internal data layout information to the host such as zone types/states and write pointers, and they do not accept non-sequential write requests; therefore, they do not need the media cache component [2][5][6]. Host Aware SMR (HA-SMR) drives are the superset of the other two types in terms of functionality, i.e., they can both handle non-sequential writes and provide the host with the internal data layout information [2][5][6]. In this paper, we choose HA-SMR drives to investigate because HA-SMR drives are expected to be the most popular among the three types in building future large-scale enterprise storage systems.

We carry out performance tests on several sample HA-SMR drives to characterize special features present in HA-SMR drives, such as the open zone issue and the non-sequential zone issue. We especially focus on their performance impacts on bursty and sustained, sequential and non-sequential write performance as well as the media cache cleaning efficiency under various workloads.

Based on our testing observations, we further propose to explore the benefit of a *host-controlled indirection buffer* (*H-Buffer*), which is a circular-log buffer consisting of a few zones that can temporarily redirect the write I/O requests. We believe that such a host-controlled indirection buffer will open a new design space for improving the I/O performance of HA-SMR drive based systems. With the additional H-Buffer, both the host and the drive can individually manage their corresponding buffers to do redirection and migration (drive controls media cache, while host controls H-Buffer). In this case, higher level applications have the flexibility to switch among all three data paths (direct writing to destination, media cache buffering, and H-Buffer buffering). So that they can combine the strengths of these data paths to improve the I/O performance by reorganizing undesirable patterns of I/O workload into preferable ones for HA-SMR drives. We take the HA-SMR open zone issue as a case study to demonstrate the potential of H-buffer.

2 Preliminaries

T10 ZBC [5] and T13 ZAC [6] standards model SMR drives as a set of *zones*, which are collections of consecutive Logical Block Addresses (LBAs). SMR bands are conceptually abstracted as *write pointer zones*, each of which has an associated *write pointer* indicating an LBA within the zone to which the next write to the zone should be targeted. Note that optionally there can be a small number of *conventional zones* which represent conventionally recorded regions on the disk (our sample drives allow ~0.2% of the reported capacity to be conventional). In this paper we mainly focus on write pointer zones, and will refer to write pointer zones as *zones* for brevity in the rest of the paper unless stated otherwise.

Applications may either follow the write pointer or not, and the corresponding write operations are defined as *sequential writes* and *non-sequential writes* respectively. For a sequential write, data is written to the write pointer, and the write pointer will be advanced accordingly. For a non-sequential write, generally HA-SMR drives will redirect it into the *media cache* and change the zone state to "non-sequential". The media cache is

one part of the disk that logs every non-sequentially written data into a self-describing journal [4]. Later, *media cache cleaning* will migrate the non-sequential data back to its original zone via a Read-Modify-Write operation. Once the cleaning is done for this zone, the write pointer will point to the next LBA of the last valid data block, and the zone state is changed back to "sequential".

There is a recommended maximum number of open zones in HA-SMR drives [5] (*open zone issue*). Before serving write I/O requests, an empty or closed zone has to be opened to get an "open zone resource" allocated for it so that the zone meta-data (e.g. write pointer and flags) that is frequently accessed and changed can persist through unexpected power loss. A write operation to a closed zone will implicitly open the zone. Since the "open zone resources" are limited, when all of them have been allocated, opening a new zone will deallocate the open zone resource of an old open zone. Such resource deallocation operation will synchronize the zone meta-data, incurring expensive disk persistence operation. As a result, if the application switches too often among a large number of zones, the write performance is expected to degrade because of frequent zone meta-data synchronization.

Similarly, HA-SMR drives also have a recommended maximum number of non-sequentially written zones [5] (*non-sequential zone issue*). Because the media cache has a limited capacity, when the non-sequential writes are too small and targeted to too many different LBAs, the media cache resources are depleted and cleaning becomes blocking, which degrades the read and write performance.

3 HA-SMR Drives Characterization

3.1 Test Setup and Basic Results

Our test environment includes several Seagate 8TB Host Aware SMR drives (Model ST8000AS0022 *prototype* firmware revision ZN03. Note that there is no mass-production non-prototype firmware available for now). The sample drives are attached to a Dell PowerEdge R420 1U server through 3Gbps SATA motherboard connectors, and have the recommended maximum numbers of 128 open zones and 8 non-sequentially written zones.

`libzbc` (branch `r04`) is used to get the geometry of the drive, monitor the write pointer location/state of the zones, and reset write pointers. Besides, we use `fio` to replay various micro-benchmark workloads to the sample drives. Preliminary tests are conducted to discover the basic internal structural information of the drive. We find that the band size is much larger for our HA-SMR sample drive (256*MiB*, which is actually the zone size) than a DM-SMR counterpart product as described in [4] (15~40*MiB*). We also observe that the average band cleaning time varies widely (1~30+ sec/zone). The reason for such variation will be investigated at length in Sec. 3.4.

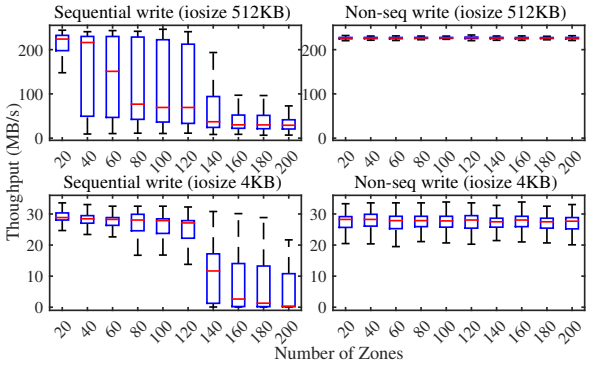

Figure 1: Throughput v.s. number of open zones.

3.2 Open Zone Issue

Although the drive specifies the recommended maximum number for the open zones, system designers still need to know exactly how the performance is going to degrade when the recommendation is not followed. In order to provide a reference point for the system designers and motivate the solutions, we conduct the following experiment to evaluate the open zone issue for both sequential and non-sequential workloads.

We use a light and bursty workload for this test, trying to isolate the open zone issue by reducing the interference from media cache cleaning. Such workload is characterized by long idle time between bursty I/O requests and can be typically found in personal computers.

The test program issues 1000 write requests in each burst in a round-robin fashion to a set of zones. The number of the zones ranges from 20 to 200. Both sequential writes and non-sequential writes are evaluated with two possible I/O request sizes: large (512*KB*) and small (4*KB*). The results are summarized as boxplots in Fig. 1. From the results we observe that for sequential writes, the performance drops clearly between 120 and 140 zones, which is consistent with the self-reported 128 open zone recommendation. For example, in the 4*KB* case the median throughput (red bar) decreases by as high as 57% from 120 to 140 zone. We have proposed a solution for such performance degradation in Sec. 5.

Surprisingly, the performance of non-sequential writes shows no significant throughput degradation as the open zone number increases. This is probably because non-sequential writes are redirected to media cache, and the self-describing journal structure ensures the consistency so that a non-sequential zone does not need to take up the "open zone resources" for reliability purpose. Therefore, the performance is no longer limited by the recommended maximum of 128 open zones.

Another counter-intuitive observation from Fig. 1 is that sequential writes do not always achieve higher bandwidth than non-sequential writes. Reasons are possibly:

1) non-sequential data blocks are actually written to the media cache which resides in higher performance outer track; 2) non-sequential data blocks are logged sequentially to the media cache, minimizing the disk arm movement; 3) each burst of non-sequential writes has not accumulated sufficient data in the media cache to trigger the blocking media cache cleaning; and 4) sequential writes performance may be affected by the open zone issue.

In conclusion, for sequential workload, it would be desirable to stay under the recommended maximum open zone number and switch gradually from one set of zones to another to avoid the performance degradation. This actually implies that the HA-SMR sample drives will provide a performance equivalent to non-shingled drives in archival storage systems, where data is sequentially written once and read many times. Besides, non-sequential writes are free of the open zone issue, therefore HA-SMR drives will have satisfactory performance for light and bursty workloads (as in personal computers), where there is enough idle time for the drive to do media cache cleaning before the next burst of writes.

3.3 Non-sequential Zone Issue

The recommended maximum non-sequential zone number indicates a constraint for the total number of zones that can be randomly written because of the media cache cleaning cost. However a legacy zone-unaware workload may easily span more zones than the suggested number (8 for our sample drives). This test is to evaluate how well the drive performs in a non-sequential workload covering a large number of zones.

In the previous test a light and bursty workload is used to reduce the media cache cleaning interference. By contrast, in this test a sustained non-sequential write workload is created which can aggregate enough data in the media cache to force the drive to do blocking media cache cleaning. Specifically, a non-sequential write workload with $256KB$ I/O size is issued into 128 and 256 zones respectively for 2 hours, with the result summarized in Fig. 2.

The non-sequential zone number plot shows that the drive performs cleaning while serving the write requests. The corresponding throughput plot exhibits bimodal performance: each time the drive starts blocking cleaning, the throughput drops from over $100MB/s$ (normal-throughput mode) to a low rate of about $0.1MB/s$ (low-throughput mode). The first low-throughput mode lasts for over 25 minutes for 128 zones and over 37 minutes for 256 zones.

Therefore, a sustained non-sequential workload will trigger blocking media cache cleaning after accumulating enough non-sequential data, and accordingly the throughput will drop seriously. A greater number of non-sequential zones leads to a longer time for the throughput to recover, indicating a larger average band cleaning time. So designers for HA-SMR systems should keep the number of non-sequential zones as few as possible to avoid this severe performance degradation. From another perspective, HA-SMR drives will fit well in a hierarchical storage architecture as the second tier where the first tier – possibly Flash – can filter out most of the non-sequential write requests. It may also work well as the first tier if a software layer can do a good job reorganizing the workload to avoid too many non-sequential zones.

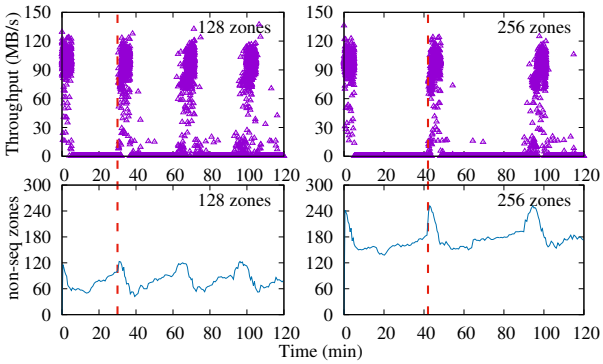

Figure 2: Throughput and number of non-sequential zones vs time when writing to 128 or 256 zones. Vertical dash lines show the correlation between the throughput and the cleaning process.

3.4 Band Cleaning Time

We noticed that the average band cleaning time varies widely in Sec. 3.1. However an accurate estimation of the band cleaning time is crucial for designers to predict the media cache cleaning efficiency and the non-sequential write performance. Therefore in this section we investigate how different factors would affect the average band cleaning time. We hypothesize that the band cleaning time depends on workload characteristics such as *zone coverage* (verified by Sec. 3.3), *update ratio*, *I/O request offset*, *I/O request number*, *I/O request size*, etc. Here update ratio (U) of a zone is defined as the proportion of data that is non-sequentially written within a zone. The I/O request offset of a zone is defined as the location where the update happens within a zone.

We design the first test to study the impact of the update ratio and the I/O request offset on the average band cleaning time.

The test program updates 100 zones with different update ratios. Once the program finishes, we measure the time it takes for the non-sequential zone number to decrease to zero and calculate the average band cleaning time by dividing the time by the number of zones that has been cleaned. Besides, four different I/O request offsets are used here: the *beginning*, *middle*, *end* and *random* offsets of the zone. Furthermore, the initial state of a zone can be either empty or full. Test results for all the combinations are summarized in Fig. 3.

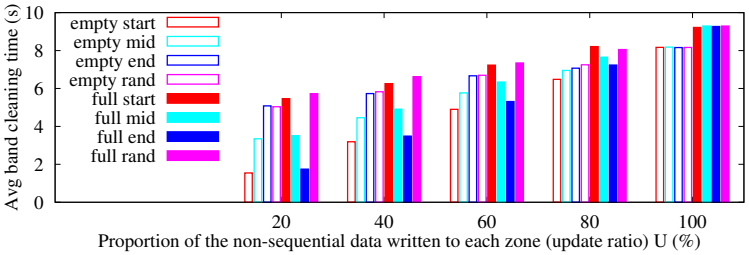

Figure 3: Average band cleaning time.

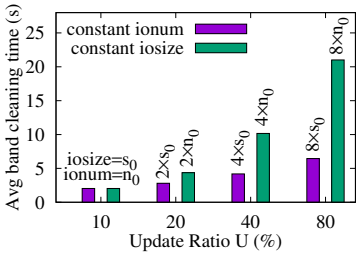

Figure 4: Compare iosize and ionum.

As update ratio U increases, the average band cleaning time goes up accordingly. When writing to an empty zone, the closer the I/O request offset is to the zone end, the longer it takes to clean that zone. The reason is that when migrating data from media cache back to original zones, the drive has to write some synthesized data to the unwritten areas before the write pointer. Therefore updating to higher LBA positions in an empty zone results in writing more synthesized data in the later cleaning process. By contrast, when writing to a full zone, the closer the I/O request offset is to the zone beginning, the longer the cleaning time is. This is because migrating data back to a full zone requires to read and rewrite valid zone data after the updated LBAs, and modifying lower LBAs causes more data to be read out and written back.

From the previous test we know that the average cleaning time goes up when more data is written to the zone. However it is unclear whether the cleaning time is the same if writing identical amount of non-sequential data with different I/O request number (ionum) and I/O request size (iosize). So in the second test, instead of increasing update ratio U by raising I/O request number n, the testing program keeps n constant but increases I/O request size s to raise U (left bars in Fig. 4). For comparison, we still carry out another set of tests, keep s the same but increase n, and get the right bars in Fig. 4. Note that we start from $U = 20\%, s_0 = 32KB, n_0 = 800$ in both cases. The zones are initially set to be empty so that the cleaning overhead is close to the effective data movement cost. It can be seen that in both cases the cleaning time increases as U grows. But if we increase I/O request size, the cleaning time grows in a slower pace than the case where we raise the I/O request number.

To sum up, the average band cleaning time is positively correlated with zone coverage, update ratio, I/O request number and I/O request size. I/O request number contributes more to the cleaning time than I/O request size does. The I/O request offset also has an influence on the cleaning time, depending on how much extra data is read/written.

4 Host-Controlled Indirection Buffer

The special performance characteristics and issues observed in the previous sections raise quite a challenge for designers of HA-SMR systems. To approach this challenge, by exploiting the unique HA-SMR hardware features and host interacting model, we propose to explore the benefit of a *Host-controlled indirection Buffer* (H-Buffer). Such H-Buffer embodies an HA-SMR three-data-path system (the *mechanism*) (Fig. 5) which can support a broad spectrum of workload separation algorithms (the *policy*). We believe such separation of mechanism and policy can eventually lead to various solutions to the performance degradation problems in HA-SMR.

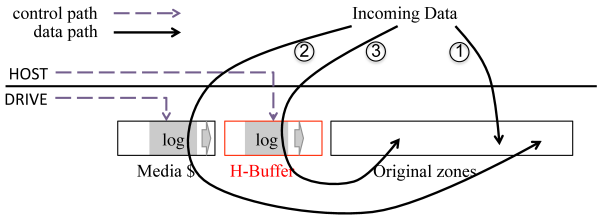

Figure 5: H-Buffer and the three data paths for HA-SMR drive.

H-Buffer Description

A small number of zones (e.g. with a recommended open zone number, or with a size comparable to media cache) of the drive are reserved as a circular buffer (H-Buffer), that can redirect incoming data. The mapping table is maintained in host memory and flushed into conventional zones occasionally, while data blocks are logged as a self-describing journal for mapping recovery. Besides, the host is responsible for migrating buffered data back to the original zone (H-Buffer cleaning).

HA-SMR Three-data-path System

As Fig. 5 shows, an HA-SMR drive natively supports data path ① and ②, in which ①, the *direct data path* represents sequential data being directly written to the destination zones; and ②, the *media cache data path* denotes non-sequential data being buffered into media cache and later migrated back to the original zones. Additionally, our H-Buffer creates a new data path ③, the *H-Buffer data path*, which is similar to the media cache data path but is controlled by the host instead of the drive. Note that HA-SMR is the only SMR model that is able to support all three data paths because it's the only one that can both handle non-sequential writes (for ②) and provide zone information to support accurate host control (for ③).

The three data paths each have their own advantages: 1) direct data path accepts sequential data without any redirection/migration overhead; 2) media cache data path can handle non-sequential data cleaning without transferring data to and from the host, which has higher data migrating bandwidth and introduces no extra computational workload for the host; 3) H-Buffer data path can leverage host's bigger memory and more powerful processors to support enhanced non-sequential data cleaning algorithm, which is expected to be more efficient; additionally, H-Buffer has the ability to redirect sequential data, which cannot be done by the media cache data path.

This proposal essentially opens a big design space for the three data path switching policies that can combine their advantages for different workloads.

5 Case Study: Open Zone Issue

We prove H-Buffer's potential by designing a simple data path switching policy (named: *open-zone-policy*) which addresses the open zone issue (Sec. 3.2). The open-zone-policy detects concurrent sequential write streams from the workload. If there are more streams than the recommended number, the policy will switch all the streams into H-Buffer and perform migration later.

The rationale is that if we redirect and log the incoming data in an H-Buffer, the drive can both avoid frequent meta-data synchronization and minimize disk arm movement. Such benefits can potentially mitigate or even surpass the H-Buffer cleaning overhead.

To evaluate the open-zone-policy, we create a micro-benchmark workload that contains 200 simultaneous sequential write streams. We assume that system reserves 100 zones for H-Buffer. The test program replays the workload to the HA-SMR drive, comparing the job completion time with and without applying the open-zone-policy. Fig. 6 summarized the test results with the I/O request sizes ranging from 4KB to 256KB. If the policy is on, the time is broken up into H-Buffer write and H-Buffer migration. We find that the total I/O time is still shorter for H-Buffer redirection than direct write even though H-Buffer is doing extra I/O operations. Therefore this H-Buffer backed open-zone-policy demonstrates H-Buffer's potential of bringing performance benefits.

6 Related Work

The closest work on SMR drive performance characterization is Skylight [4], which uses both software and hardware approaches to uncover the internal structure of a DM-SMR drive. By contrast, we investigate unique features of HA-SMR drives that do not exist in DM-SMR drives, such as the open zone issue. Besides, we exploit the richer ZBC API of the HA-SMR drives to aid the interpretation of the performance results.

For SMR drive data handling, previous works mainly focus on data layout design [7][8], indirection systems

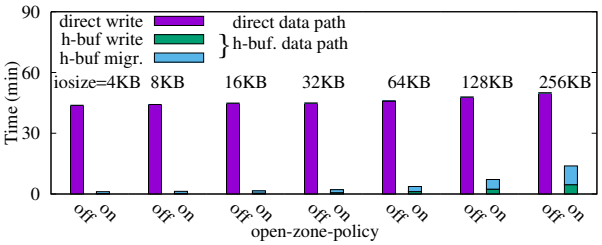

Figure 6: H-Buffer helps addressing the open zone issue.

[9][10], HM-SMR operation models [11], etc. However, our paper differs from previous works in that we emphasize the unique three data paths for the HA-SMR model, management separation between the host and the drive, and how to combine each of their advantages.

7 Conclusion and Future Work

We present a study on HA-SMR drives by carrying out extensive testing on several HA-SMR sample drives. Specifically, we investigate performance impact of the open zone and non-sequential zone issues, as well as the factors that influence the media cache cleaning efficiency. Based on the empirical observations, we summarize the system implications and propose a novel H-Buffer that can potentially improve the I/O performance of HA-SMR drives. In our future work, we plan to investigate different data path switching policy so that HA-SMR drives can be used to construct large-scale storage systems that support various workloads.

Acknowledgments

This work is partially supported by the following NSF awards: 1439622, 1305237, 1421913, 1217569 and 1115471. We thank the anonymous reviewers, Timothy Feldman, Andrew Kowles, and our shepherd, Pallavi Joshi, for their feedback.

References

[1] Roger Wood, Mason Williams, Aleksandar Kavcic, and Jim Miles. The feasibility of magnetic recording at 10 terabits per square inch on conventional media. *IEEE Transactions on Magnetics.*, 45(2):917–923, 2009.

[2] Timothy R. Feldman. Host aware smr. `http://open-zfs.org/w/images/2/2a/Host-Aware_SMR-Tim_Feldman.pdf`, OpenZFS Develop Summit, San Francisco, November 2014.

[3] Tim Feldman and Garth Gibson. Shingled magnetic recording areal density increase requires new data management. *USENIX ;login:*, 38(3), 2013.

[4] Abutalib Aghayev and Peter Desnoyers. Skylight–a window on shingled disk operation. In *Proc. FAST'15, Santa Clara, CA, USA.*

[5] INCITS T10 Technical Committee. Information technology - zoned block commands (zbc). 2015.

[6] INCITS T13 Technical Committee. Zoned-device ata command set (zac) working draft.

[7] Weiping He and David HC Du. Novel address mappings for shingled write disks. In *Proc. HotStorage'14,Philadelphia, PA, USA.*

[8] Ahmed Amer, Darrell DE Long, Ethan L Miller, J-F Paris, and SJT Schwarz. Design issues for a shingled write disk system. In *Proc. MSST'10, Incline Village, NV, USA.*

[9] Yuval Cassuto, Marco AA Sanvido, Cyril Guyot, David R Hall, and Zvonimir Z Bandic. Indirection systems for shingled-recording disk drives. In *MSST'10, Incline Village, NV, USA.*

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

[11] Saurabh Kadekodi, Swapnil Pimpale, and Garth A Gibson. Caveat-scriptor: write anywhere shingled disks. In *Proc. HotStorage'15, Santa Clara, CA, USA.*