國立台灣科技大學

電子工程系

碩士學位論文

一種減少疊瓦式硬碟的讀寫頭移動之方法

A Method to Reduce Read/Write Head Movement for SMR Disks

研究生：林品逸

學　　號：M11102163

指導教授：吳晉賢博士

中華民國 114 年 01 月 07 日

M11102163

碩士學位論文指導教授推薦書

Master's Thesis Recommendation Form

系所： Department/Graduate Institute	電子工程系 Department of Electronic and Computer Engineering
姓名： Name	林品逸 Lin,Pin-Yi
論文題目： (Thesis Title)	一種減少疊瓦式硬碟的讀寫頭移動之方法 A Method to Reduce Read/Write Head Movement for SMR Disks

係由本人指導撰述，同意提付審查。

This is to certify that the thesis submitted by the student named above, has been written under my supervision. I hereby approve this thesis to be applied for examination.

指導教授簽章：
Advisor's Signature

共同指導教授簽章（如有）：
Co-advisor's Signature (if any)

日期：
Date(yyyy/mm/dd) 114 / 1 / 7

M11102163

碩士學位考試委員審定書

Qualification Form by Master's Degree Examination Committee

系所： Department/Graduate Institute	電子工程系 Department of Electronic and Computer Engineering
姓名： Name	林品逸 Lin,Pin-Yi
論文題目： (Thesis Title)	一種減少疊瓦式硬碟的讀寫頭移動之方法 A Method to Reduce Read/Write Head Movement for SMR Disks

經本委員會審定通過，特此證明。

This is to certify that the thesis submitted by the student named above, is qualified and approved by the Examination Committee.

學位考試委員會

Degree Examination Committee

委員簽章：

Member's Signatures

張原豪　謝仁偉

陳奕伸　吳晉賢

指導教授簽章：

Advisor's Signature　吳晉賢

共同指導教授簽章（如有）：

Co-advisor's Signature (if any)

系所（學程）主任（所長）簽章：

Department/Study Program/Graduate Institute Chair's Signature　林淵翔

日期：

Date(yyyy/mm/dd)　114/1/17

中文摘要

爲了滿足日益增長的數據存儲需求，瓦片式磁記錄 (Shingled Magnetic Recording, SMR) 硬碟被採用作爲一種高密度的非揮發性儲存介質，與傳統硬碟相比，它在儲存容量和成本方面具有顯著的優勢。然而，低效的讀-修改-寫（Read-Modify-Write, RMW）操作與巨量的隨機資料存取對 SMR 硬碟造成大量的 I/O time。本研究通過結合 SMR 的大容量存儲優點與 CMR 的就地更新特性，同時利用多顆硬碟的存取方式達到平行處理，以及動態 I/O 分類使系統能夠依據 I/O 請求進行分類並與優化的 C-Look 排班法相結合，有效的降低了磁頭移動時間與響應時間並提升了系統整體性能。與基線相比，我們的方法在磁頭移動時間上平均減少了 133.49%，在響應時間方面平均降低了 21.97%。

ABSTRACT

To address the growing demand for data storage, Shingled Magnetic Recording (SMR) hard disks have been adopted as a high-density, non-volatile storage medium that offers significant storage capacity and cost advantages over traditional hard disks. However, inefficient Read-Modify-Write (RMW) operations and huge amount of random data accesses cause a large amount of I/O time for SMR hard disks. We combine the advantages of large-capacity storage of SMR and the in-place updating characteristics of CMR, and at the same time utilizing the access method of multiple hard disks to achieve parallel processing, as well as dynamic I/O classification to enable the system to classify I/O requests according to the I/O requests, and combining with the optimized C-Look scheduling method, the system effectively reduces the head movement time and response time and improves the overall system performance. Compared to the baseline, our approach reduced head movement time by an average of 133.49% and response time by an average of 21.97%.

Content

Figure Directory

Table Directory

Chapter 1 Introduction

Faced with the growing challenge of increasing performance and capacity requirements for data storage, Shingled Magnetic Recording (SMR) technology, with its superior storage density characteristics, provides an effective solution to address the need for high-capacity storage. However, the most significant new challenge that comes with the adoption of SMR technology is the Read-Modify-Write (RMW) operation that must be performed when updating data, which requires rewriting the entire data block, resulting in frequent head movements and extended response times. In traditional hard disk drives (HDDs), the response time is most affected by the seek time, i.e., the head movement time. Therefore, this study focuses on how to effectively reduce the head movement by developing advanced storage allocation strategies and scheduling algorithms on the system design to reduce the seek time, thus improving the access efficiency of the HDDs, and providing a more optimized solution for handling large-scale data.

We proposes an innovative hybrid access framework that aims to combine the high storage capacity of SMR with the high performance in-place update strategy of CMR and the adaptive head scheduling method to reduce the head movement time, and based on this, the application of parallel processing through the access strategy of multiple hard

disks improves the system performance. A hybrid architecture with adaptive dynamic I/O classification and adaptive C-LOOK scheduling algorithms are introduced into the system architecture, and these strategies not only enhance the access efficiency, but also significantly reduce the overall response time of the system. This study experimentally modifies the Skylight simulator [1] to implement the proposed method and demonstrates that it is better than the unused method. Compared to the baseline, our approach reduced head movement time by an average of 133.49% and response time by an average of 21.97%.

The remaining chapters of this paper are organized as follows: section **??** introduces the relevant background and related work, section 3 describes the motivation of the study, section 4 describes our proposed methodology, section 5 provides an experimental comparison with baseline,and finally section 6.

Chapter 2 Background and Related Work

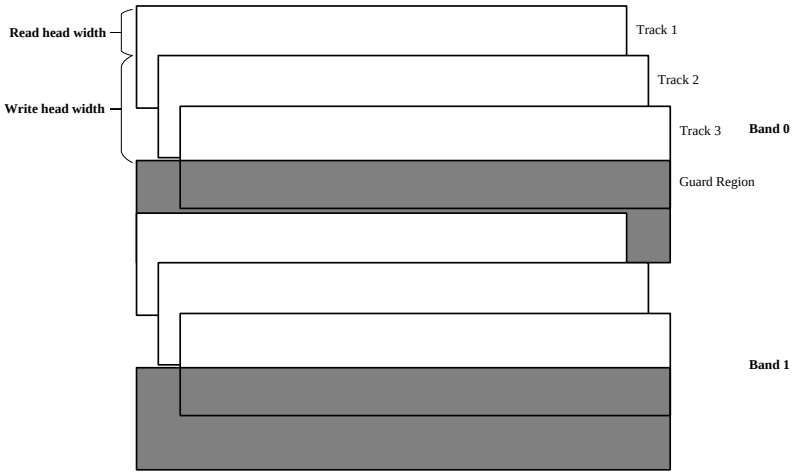

Figure 2.1: Guard region

2.1 Development of SMR disk technology

Shingled Magnetic Recording (SMR) technology is an important innovation to increase storage density [1–3].It is designed to increase the storage capacity by overlapping the magnetic tracks [4]. Compared to Conventional Magnetic Recording (CMR), SMR disks utilize a narrower track alignment to achieve higher densities. However, since the

write heads are wider than the read heads, direct overwriting of data may damage the data on the neighboring tracks. SMR is composed of several bands, each containing multiple overlapping tracks protected by guard tracks, allowing independent operation of each band without interference, as shown in Fig.2.1.

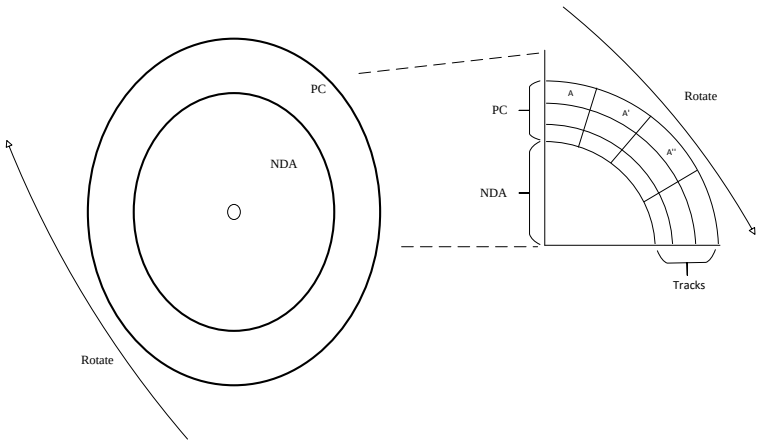

Figure 2.2: PC-NDA

2.2 PC and NDA characters and data management

SMR disks are designed with two main areas. The Persistent Cache (PC) is usually located at the Outer Diameter (OD) of the disk and is used for temporary storage of random or hot data [4,5]. Native Data Area (NDA) is usually located at the Inner Diameter (ID) of the disk and is used for storing sequential or cold data [6,7]. Persistent Cache (PC) is

used to temporarily store randomly written data, reducing the frequency of Native Data Area (NDA) operations and the number of Read-Merge/Modify-Write (RMW) operations in the system. PC can effectively convert random I/O to sequential write, and when the data accumulates to a certain amount, Garbage Collection (GC) will be triggered to move the valid data to NDA, as shown in Fig. 2.2. NDA is more suitable for storing cold data or large-capacity data written sequentially to reduce the number of head movements and increase capacity utilization. The temporary storage function of PC helps to reduce the number of random writes directly to NDA, which further reduces the number and burden of RMW operations. However, when the PC runs out of space and needs to be cleaned up, the system must perform an RMW operation. This process merges or modifies the data in the PC with the related data in the NDA and rewrites it back. Although the RMW operation maintains data consistency, it causes the Write Amplify (WA) phenomenon and a significant amount of head movement, which further reduces system performance.

2.3 Management modes of SMR disks

SMR disks are classified into three management modes according to different application requirements: Due to the rapidly growing demand for high-capacity storage, three types of SMR disks have been developed: SMR Disks Host-Managed SMR (HM-SMR)

disks, Host-Aware SMR (HA-SMR) disks. and Device-Managed SMR (DM-SMR) disks to cope with the characteristics of SMR. In recent years, several studies have focused on improving the performance of these three types of SMR disks. In addition, the concept of hybrid SMR disks has also been proposed to improve system performance by combining the advantages of the hybrid SMR/CMR format and log structure updates [8].

2.3.1 Host-Managed SMR (HM-SMR)

In HM-SMR, all I/O operations are controlled by the host, and several studies have proposed different schemes to enhance the performance of HM-SMR disks. HMSS [9] sense the SMR technique at the block level and designing a dynamic address mapping table and efficient B+ tree management combined with a small portion of legacy zones as data cache, the random I/O is effectively transformed into sequential I/O, thus significantly improving the performance of host-managed SMR disks. LaDy [10] realizes data de-duplication on SMR disks through locality-aware de-duplication technology, taking into account the impact of de-duplication on data locality and read performance. Data locality is maintained by selectively writing duplicate data, which improves the read performance of the SMR drive, and the garbage collection overhead is reduced by the GC de-duplicator, which ultimately reduces the response time by up to 87.3% in the experiment.

2.3.2 Host-Aware SMR (HA-SMR)

HA-SMR allows hosts and disks to manage I/O together. some studies have proposed different schemes to enhance the performance of HA-SMR disks. Previous research [11] has improved the stability and predictability of HA-SMR drives by designing a user-level file system for host-aware shingle magnetic recording (HA-SMR) drives that actively manages data in persistent caches and cleans up the caches at the right time to significantly reduce the number of high-latency requests. By integrating persistent memory (PM) and traditional disks, TPFS [12] utilizes the designed data placement strategy and data migration mechanism to achieve a file system that is close to the performance of PM under different access modes. At the same time, TPFS utilizes dynamic thresholds to balance read and write traffic, and maximizes the sequential bandwidth of disks through a group migration mechanism to effectively improve system performance and optimize storage resource utilization.

2.3.3 Device Managed SMR (DM-SMR)

In DM-SMR, disks manage I/O automatically. KFR [13] reduces cleanup overhead by using a delayed write policy to identify disk space that does not affect neighboring

areas, delaying the PC cleanup process and cleaning up only a small part of the PC space. MU-RMW [14] reduces unnecessary RMW operations in embedded Flash and SMR disk systems through centralized address mapping and lazy RMW reclamation policy. MU-RMW avoids mixing data from different SMR sectors in the same Flash block, thus reducing system write amplification and cleanup overhead and improving overall system performance. By using rewritable regions in the SMR translation layer, uCache [15] reduces the garbage collection overhead caused by off-site write operations by taking advantage of the spatial and temporal locality of the workload. uCache upgrades the data blocks to the rewritable regions, so that subsequent writes can be performed in-place, thus reducing the garbage collection loop. This reduces the frequency of garbage collection cycles and improves system performance. Tiler [16] proposes to divide the SMR disk into autonomous regions (ARs). Each AR contains an SMR band and a corresponding PC, and when updating the SMR band in an AR, the PC in that AR is responsible for RMW operations. The PCs in the ARs are responsible for handling RMW operations when updating SMR bands in the ARs, and the PC cleanup is delayed by a delayed write policy to reduce the system load and seek time.

2.3.4 Hybrid-SMR (Hybrid-SMR)

The traditional SMR disk architecture consists of SMR regions, persistent caches (PCs), and PC mapping tables [5]. The advantage of this design is that it simplifies the management of PCs and SMR regions, thus alleviating the RMW operation problem caused by the out-place update process of SMR disks. However, due to the large amount of temporary data piling up in the PC, it may lead to a long blocking time when performing garbage collection. Especially when the data is rewritten from the PC to the SMR area, the frequent head movement will further degrade the system performance.

To improve this situation, a previous study FluidSMR [17] explored a hybrid architecture that combines the SMR and CMR formats. FluidSMR divides disks into multiple H-partitions, each of which contains a CMR part, an SMR part, and its corresponding PC. When the data in an H-partition is in the CMR part, the system can in-place update; if the data is in the SMR part, it will be written to the PC in that H-partition first. In addition, FluidSMR dynamically adjusts the size of the CMR and SMR areas according to actual usage to further optimize data access performance.

2.4 Parallel Processing Applications

In modern data storage systems, the core strategy for improving performance is the multiple hard disks cooperation and parallel processing. By distributing I/O requests across multiple disks in parallel, the system can effectively reduce the range and number of head movements, which in turn reduces access latency and improves performance. The use of multiple hard disks allows for simultaneous processing of large amounts of data reads and writes, preventing a single disk from becoming a bottleneck and triggering large head movement times. In addition, storing data blocks on separate disks further improves data access speed and system stability.

Parallel processing is a key technique in multi-bay HDD design. In RAID4SMR [18], the authors propose a RAID 4SMR architecture consisting of three SMR disks, a data hard disk (HDD), and a parity drive. The data HDD stores updated data to reduce garbage collection frequency. I/O requests are distributed using polling or shortest-path algorithms, enabling simultaneous operations across multiple SMR disks, which significantly improves system performance and stability.

2.5 Disk scheduler

In the field of disk scheduling optimization, many researchers have previously investigated how to reduce the head travel time and have conducted various evaluations to verify the performance of different algorithms. Traditional disk scheduling algorithms, such as FCFS, SSTF, SCAN, LOOK, etc., are generally used in many systems [19,20]. These algorithms mainly focus on reducing the head travel distance to improve the overall I/O performance. Shankar et al. [21] provides an in-depth comparison of various scheduling algorithms, emphasizing the benefits of different algorithms in terms of workload and actual performance. Mishra et al. [22] proposed an improved version of FCFS (IFCFS), which aims to reduce the total distance traveled by the head and demonstrates that the improvement can significantly reduce the search time. Meanwhile, Saha et al. [23] also proposed a new heuristic algorithm specifically designed to reduce the number of head moves in order to improve throughput and system performance.

In comparison, the C-LOOK algorithm demonstrates better efficiency in most work environments.According to several studies [24,25], the C-LOOK algorithm effectively reduces the total number of head moves. The C-LOOK algorithm can effectively reduce the total number of head movements and minimize the search time by avoiding unnecessary head movements. These optimizations allow C-LOOK to provide the least head

movement time under different loads, and it demonstrates extremely stable and excellent performance especially in the case of large number of I/O requests [26]. As a result, C-LOOK has become the algorithm of choice for most high-performance systems and is widely used in modern storage systems to maximize performance.

Chapter 3 Motivation

In the conventional SMR architecture, the disk stores update/write operations in the PC using an append mode in the rotational direction [8] to mitigate the negative impact of RMW. However, when the PC space reaches a certain threshold or becomes full, the disk writes the valid data cached in the PC back to the NDA in a single band [1,5] to free up PC space. This process triggers RMW operations, affecting all data stored on subsequent tracks. Specifically, the frequent back-and-forth movement of the read-write head during data writing from the PC to the NDA is time-consuming and significantly reduces system performance.The issue of track movement becomes particularly severe when head movement strategies are not optimized, as head movements increase seek time, which is a critical factor in determining data transfer efficiency. Seek time has a lasting impact on overall access times, hindering fast data retrieval and storage. This impact is especially detrimental to data-intensive applications that require real-time data access, potentially becoming the main bottleneck for system performance.

Based on the above previous studies related to the reduction of head movement, although many studies have been conducted on head scheduling optimization, these studies tend to focus on improving the physical strategy of head movement, but lack an in-depth

research on the design of the I/O management layer, especially on how to further reduce the head movement time through I/O management optimization. Therefore, the motivation of this study is to explore how to combine I/O management design with head scheduling optimization to achieve more effective seek time reduction.

Chapter 4 A Method to Reduce Read/Write Head Movement for SMR Disks

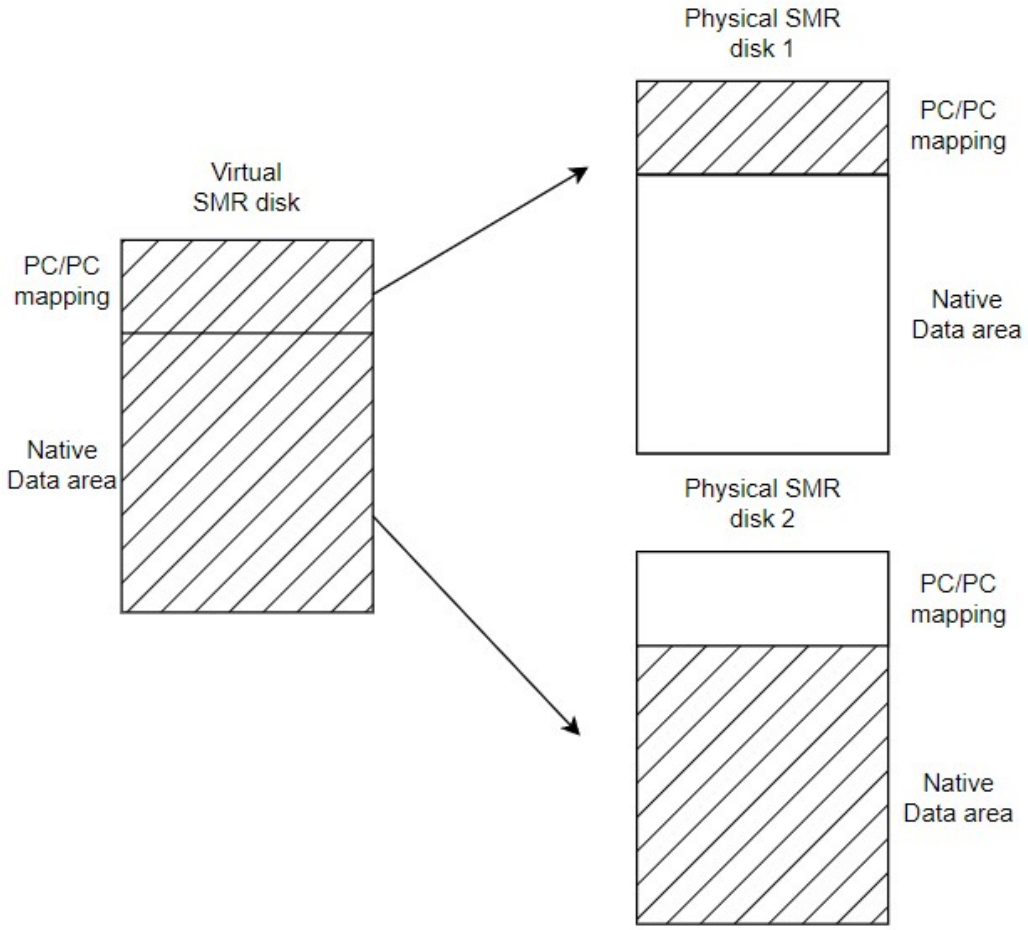

Figure 4.1: Distribute SMR

4.1 Data distribute placement

In traditional SMR disks, all data is typically stored in the PC/PC mapping area or the native data area (NDA). However, this architecture often results in frequent movements of the read/write head between the PC/PC mapping area and the NDA during read/write requests, increasing the burden on the read/write head and adversely affecting system performance.

To address these challenges, we propose a method to limit the movement of the read/write head to a specific range for a certain period, thereby minimizing the impact of frequent head movements. Additionally, this method assigns random data and sequential data separately to the PC/PC mapping area and the NDA, respectively, further optimizing data access efficiency. Our method can also be applied to multiple SMR disks by distributing the PC/PC mapping area and its corresponding NDA of an original SMR disk across different SMR disks, enabling parallel processing among disks and significantly improving system performance.

For instance, as shown in Fig. 4.1, assume that the PC/PC mapping area of a virtual SMR disk is stored on physical SMR disk 1, while its corresponding NDA is stored on physical SMR disk 2. Under this architecture, when I/O requests are executed simulta-

neously on the virtual SMR disk, there is a great opportunity to constrain the read/write head movement within specific regions, thereby enhancing performance through parallelization. Furthermore, when the virtual SMR disk requires read-modify-write (RMW) operations, our method allows simultaneous reading of the latest data from the PC/PC mapping area on physical SMR disk 1 and the NDA on physical SMR disk 2. The latest data is then merged, and the merged data is written back to the PC/PC mapping area on physical SMR disk 1 or the corresponding NDA on physical SMR disk 2, thereby significantly reducing the frequency of head movements.

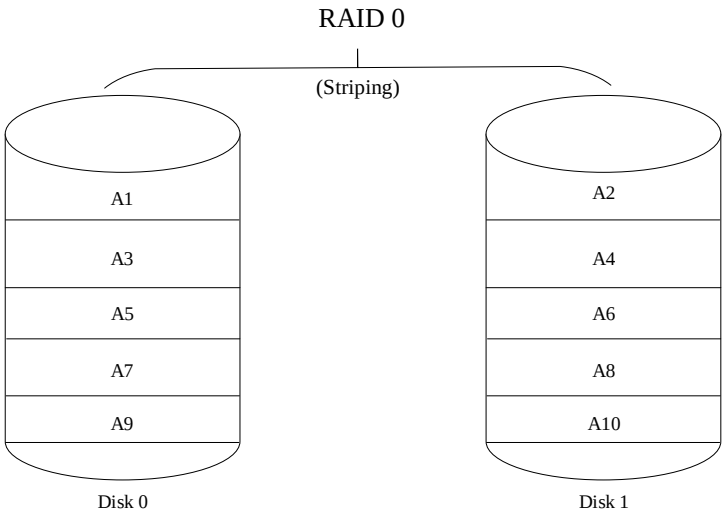

Figure 4.2: Striping

Unlike the striping technique employed by traditional RAID 0 systems (as illustrated in Fig. 4.2), our approach is specifically designed to address the unique characteristics of SMR disks. It incorporates constraints on head movement, effectively limiting the range of read/write head operations to reduce performance bottlenecks caused by frequent head movements.

4.2 I/O Scheduling

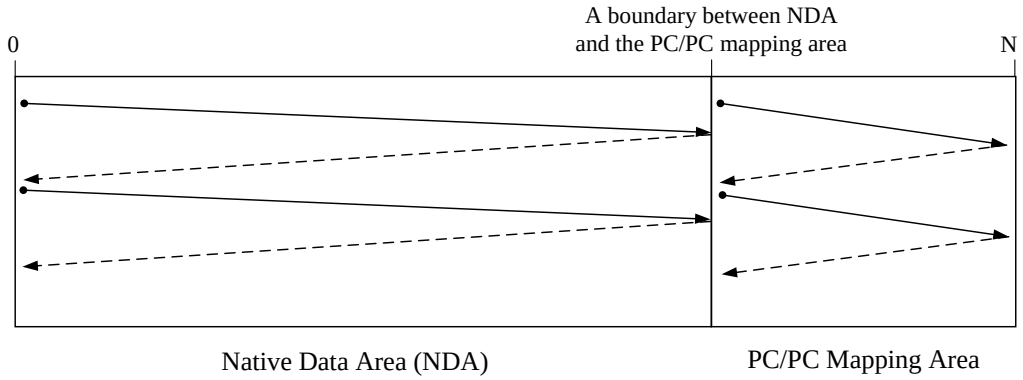

Figure 4.3: A boundary between NDA and between PC and PC/PC Mapping Area

To reduce the read/write head movement as much as possible, I/O scheduling is needed to limit the movement of the read/write head to a specific range for a specific period of time. That is, when the read/write head is moved to a specific range (e.g., the PC/PC mapping area), I/O scheduling should specifically schedule specific I/O requests in

this specific range (e.g., the PC/PC mapping area) so that the movement of the read/write head should be as small as possible. In fact, I/O requests can be divided into two parts: access the PC/PC mapping area and access NDA. When an I/O request is performed, we will try to schedule the subsequent I/O requests to access the same or close space (e.g., the same/adjacent bands in the PC/PC mapping area or NDA) and minimize the frequent movement of the read/write head between the PC/PC mapping area and NDA. For example, as shown in Fig. 4.3, when I/O requests that access NDA or the the PC/PC mapping area are performed, a boundary between them can keep the subsequent I/O requests in their specific range as much as possible by directly reversing its direction. However, when I/O scheduling perform many I/O requests to access the same or close space, the starvation of specific I/O requests during I/O scheduling will arise and will be handled in the following Section 4.3.

4.3 A Revised C-LOOK Algorithm

I/O scheduling should cooperate with the revised C-LOOK algorithm and let the read/write head move in the same or close access space (e.g., the same/adjacent bands in the PC/PC mapping area or NDA) as much as possible. In a traditional C-LOOK algorithm, it can efficiently sort I/O requests by scanning from one end of a disk to the other after

performing the final I/O request, and then jump directly to the next round of new I/O requests in a queue without reverse scanning. Unlike the traditional C-LOOK algorithm, the revised C-LOOK algorithm will sort two groups of I/O requests that access the PC/PC mapping area and NDA, and just scan from one end of the PC/PC mapping area and NDA to the other within each group to prevent starvation of specific I/O requests or delay the setup of wait times for I/O processing.The revised C-Look has been divided into two conditions.

Unlike the traditional C-LOOK algorithm, the revised C-LOOK algorithm divides the accessed I/O requests into two groups (PC/PC mapping area and NDA mapping area), and the head needs to prioritize the I/Os in the current area and move to the other area for accessing after accessing to minimize the read/write head's moving time. When the head is moving, the revised C-LOOK algorithm sets the upper limit of accesses between groups to avoid starvation of specific I/O requests or delayed I/O processing waiting time, and we allow new I/O requests to be added to the current moving path to improve efficiency for better accesses.

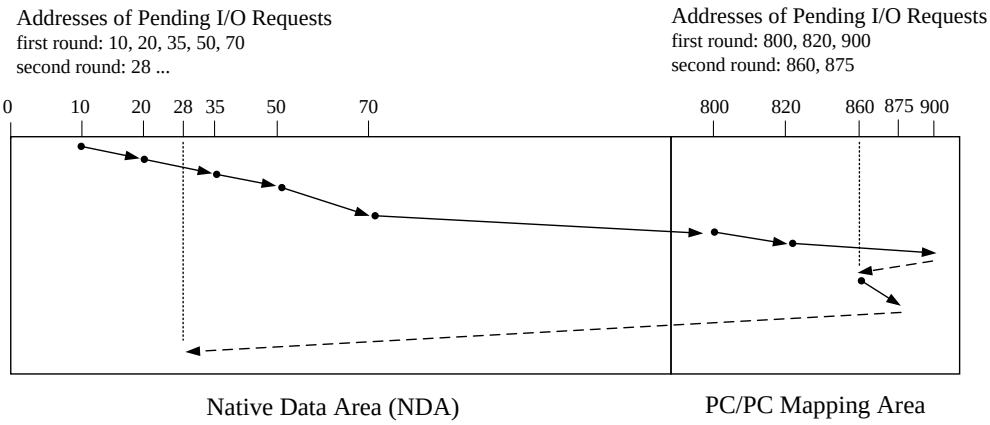

Figure 4.4: An Example of Condition 1: Revised C-LOOK Algorithm

4.3.1 First condition

When accessing I/O, prioritize the I/O of the current area. For example, as shown in Fig. 4.4, assume that the first round of pending I/O requests of NDA will access 10, 20, 35, 50 and 70; meanwhile the first round of pending I/O requests of PC/PC mapping area will access 800, 820 and 900. After the revised C-LOOK algorithm sort two groups of I/O requests that access the PC/PC mapping area and NDA, the magnetic head may move from 10, 20, 35, 50 and 70 within a sorted group of I/O requests that access NDA and then move from 800, 820 and 900 within another sorted group of I/O requests that access the PC/PC mapping area. Then, since the magnetic head is in the PC/PC mapping area, the second round of pending I/O requests that access the PC/PC mapping area will be processed first according to their sorted addresses (e.g., 860 and 875). After the I/O

request in 875 is performed, the magnetic head will move to 28 for the second round of pending I/O requests of NDA. Furthermore, when the magnetic head moves forward to process I/O requests, we allow new I/O requests to be added to the current moving route to improve efficiency.

4.3.2 Second condition

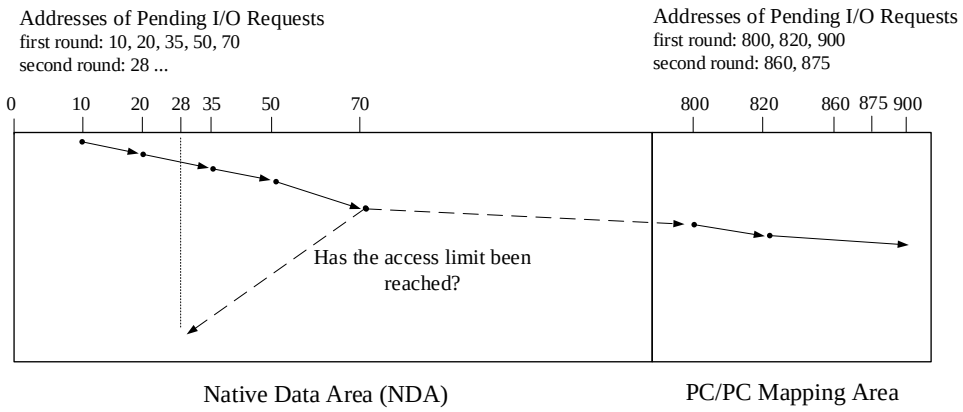

Figure 4.5: An Example of Condition 2: I/O Bandwidth Algorithm

When accessing I/O, set the upper limit of access amount for each area, if the upper limit of access amount is reached, then the head will shift to another area for accessing. For

example, as shown in Fig. 4.5, assume that the first pending I/O request in the NDA will access 10, 20, 35, 50, and 70, while the first pending I/O request in the PC/PC mapping area will access 800, 820, and 900. The revised C-LOOK algorithm head may move from the 10, 20, 35, 50, and 70 position in the sorted group of I/O requests accessing the NDA. 70, if the head moves to position 70 in the NDA partition, the accesses in that partition (NDA) have reached the upper limit and the head must move to another partition (PC) to process the I/O request, there is also an upper limit set for accesses in the PC area.

4.4 Data Consistency

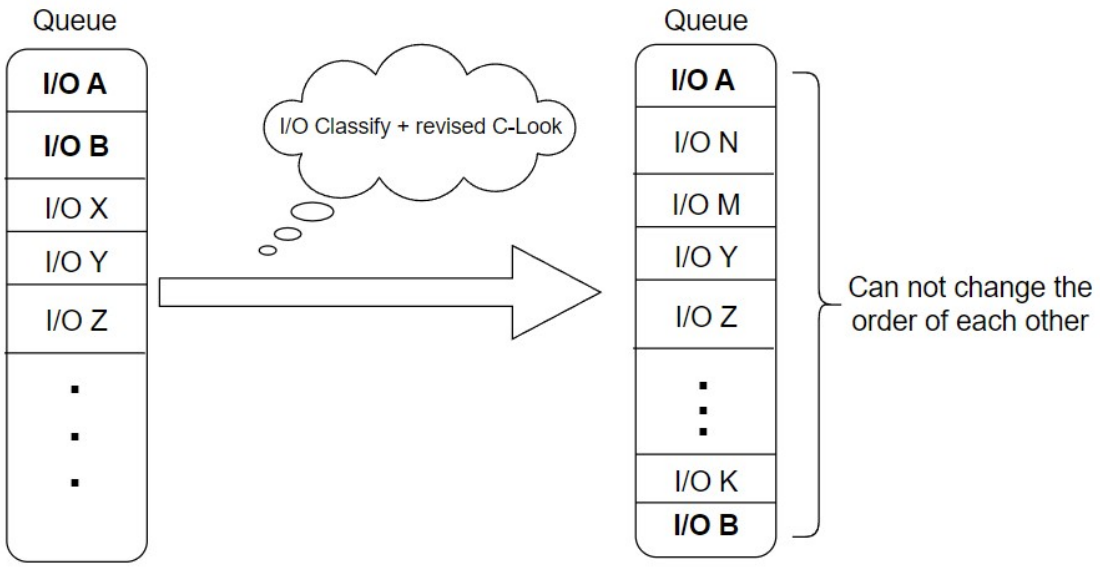

Figure 4.6: Data Consistency

The purpose of this design is to solve the data consistency problem. In this system, I/O classification and allocation across multiple hard disks affects the data accuracy of each I/O during read and write, so it is necessary to ensure that the integrity of the sequence of I/O operations in each queue is maintained. First of all, I/Os entering the hard disk queue may cause I/O sequences to be shifted due to I/O classification or revised C-Look. If there are multiple I/Os in the queue waiting for access, and the order of I/O A and I/O B cannot be changed, in order to avoid data criticality when accessing data, it is necessary to make sure that the order between I/O A and I/O B will not be changed when doing I/O categorization or adaptive C-Look, i.e., the order between them can be changed as long as there is no criticality problem, i.e., the order between them can be adjusted. As long as the order between the two remains unchanged, no data criticality problem will occur, i.e., the I/O between the two can be reduced or increased without data criticality, as shown as Fig. 4.6

Chapter 5 Performance Evaluation

5.1 Experimental Setup

Table 5-1: System Parameters

Parameter	**Value**
Drive type	SMR
Capacity	5TB
Number of SMR	4
RPM	5980
PC Size	20GB
Band Size	18–36GB
Size of a Sector	4KB
Number of Total Bands	100
Number Tracks per Band	20
Number Tracks per Zone	6000
Number Mapping Entries	200000

The specification parameters for SMR disks are shown in Table 5-1. We modified the

Skylight [1] simulator, which is a recognized SMR simulation tool, for performance evaluation.The simulated SMR disks are based on the following specifications: 5TB capacity, configured with 100 tapes containing 20 tracks each. We simulated the operation of 4 shared SMR disks with mixed workloads. On the host side, we simulated a block-level I/O scheduler and compared the proposed methodology with the baseline.We compare the proposed method with the baseline method.

Table 5-2: Mixed Workloads

Mix	**Workload**
Mix1	hm_0, prn_0, prn_1, proj_0
Mix2	wdev_0, prxy_0, src1_2, usr_0
Mix3	hm_0, prn_1, wdev_0, src1_2
Mix4	prn_0, prxy_0, stg_0, web_0
Mix5	stg_0, proj_0, src1_2, prxy_0
Mix6	web_0, usr_0, hm_0, wdev_0
Mix7	src1_2, prn_1, wdev_0, web_0

We selected ten actual workload trace files from Microsoft Research Cambridge (MSR) [27], from which we selected a combination of four workloads to generate seven hybrid workloads as shown in 5-2. The characteristics include the total number of re-

Table 5-3: Workload Characteristics

Trace	Total Number of Requests	Write Ratio (%)	Update Ratio (%)	Average Write Size (KB)	Average Read Size (KB)	Random I/O Ratio (%)	Sequential I/O Ratio (%)
hm_0	3,993,317	65%	60%	8.13	7.80	36.33	63.67
prn_0	1,111,160	11%	82%	23.01	14.11	43.54	56.46
prn_1	1,020,044	75%	77%	22.47	9.62	68.21	31.79
proj_0	4,224,525	88%	81%	32.35	8.75	36.50	63.50
wdev_0	1,143,262	80%	76%	12.60	8.40	45.06	54.94
prxy_0	1,111,160	97%	96%	6.81	5.31	42.04	57.96
src1_2	1,907,774	75%	68%	19.32	33.53	24.76	75.24
usr_0	2,237,890	38%	10%	1.83	9.89	38.13	61.87
stg_0	2,030,916	85%	81%	25.11	9.42	27.70	72.30
web_0	2,029,946	70%	67%	6.13	9.74	35.08	64.92

quests, write ratio and update ratio as shown in 5-3. The write ratio is the ratio of the number of write requests to the total number of requests, the update ratio is the ratio of the number of update requests to the number of write requests, the random I/O ratio is the ratio of the number of random data requests to the total number of requests, and the sequential I/O ratio is the ratio of the number of sequential requests to the total number of requests.

- Baseline methodology: Using Skylight [1] and SMR modeling [5], different SMRs receive I/O requests from different users, ensuring that each SMR disk receives the same number of I/O requests from different data streams.Ensure that each SMR disk receives the same number of I/O requests from different data streams.

5.2 Experimental Results

5.2.1 Average Seek Time

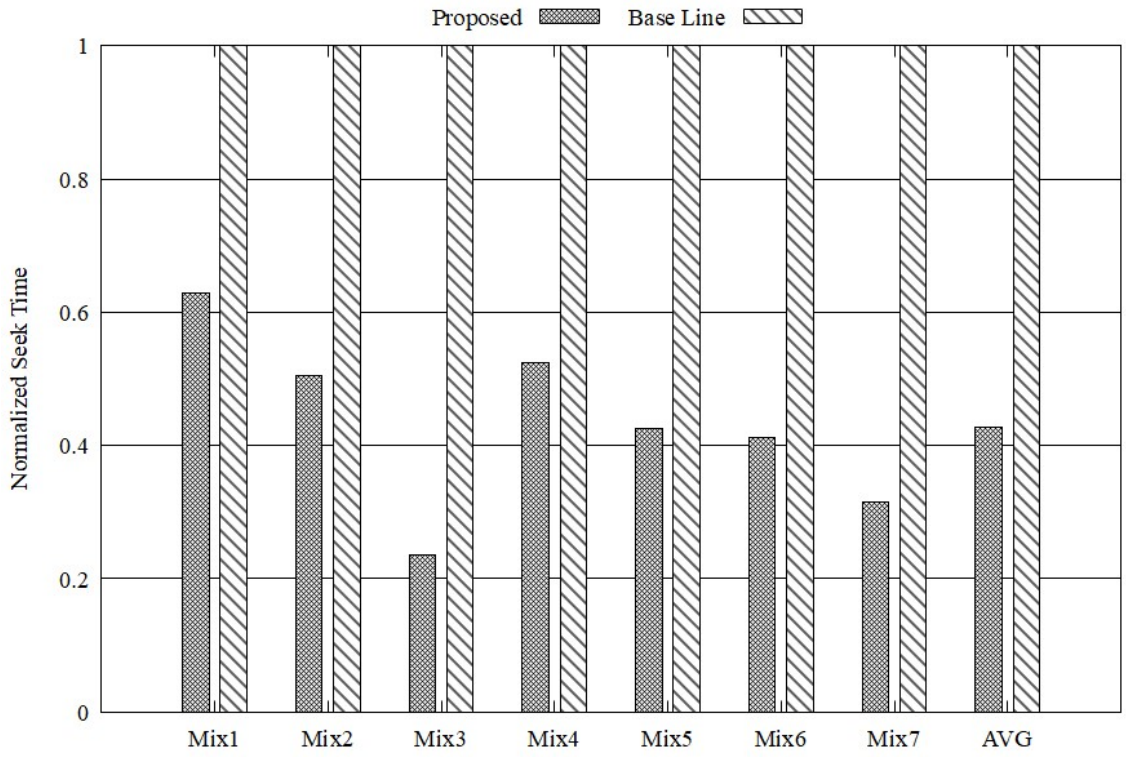

Figure 5.1: Average Seek Time

As shown as Fig. 5.1 the average seek time of our method compared with each Mix of Baseline.The average seek time is defined as the total seek time required to complete the requests of all SMRs divided by 4 (the number of SMRs).The result shows that our method can reduce the average seek time by about 133.49%.The graph shows that Mix3, Mix7 seek time is lower than Baseline, because the ratio of random data in prn_1 in trace is higher.Our method minimizes the track movement time by sorting the random data into

PCs and doing batch processing.

5.2.2 Average Response Time

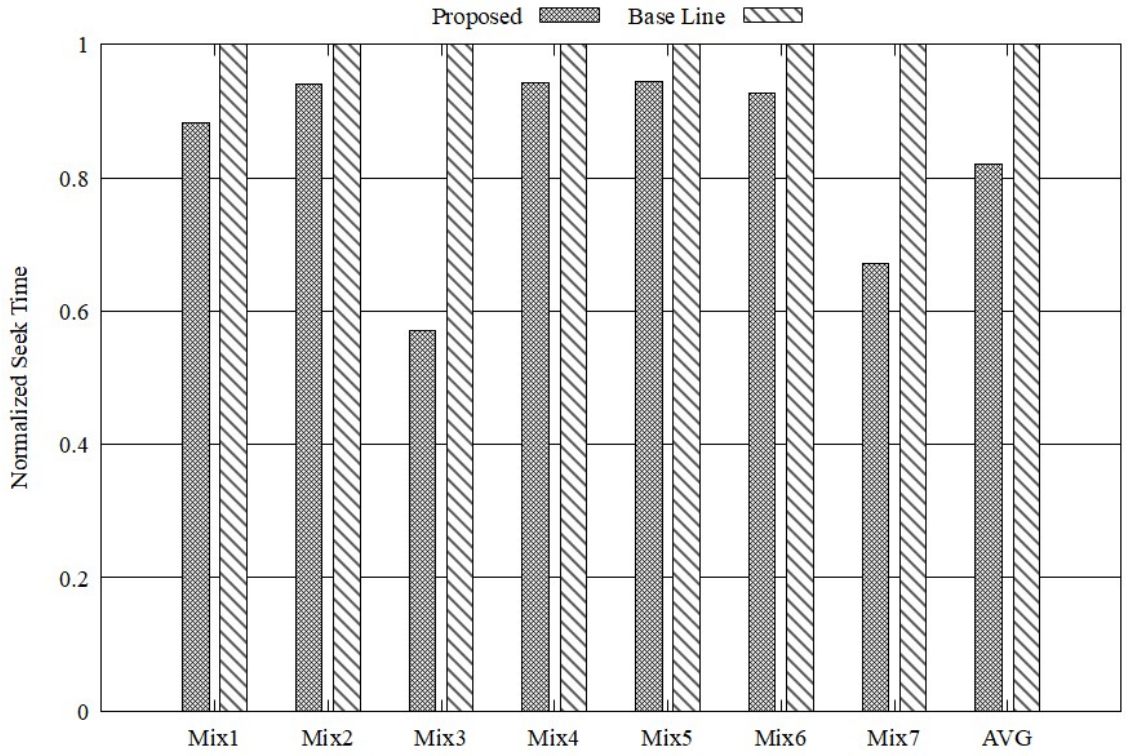

Figure 5.2: Average Response Time

As shown as Fig. 5.2 the average response time of our method compared with trace in each Mix of Baseline.The average response time is defined as the total response time required to complete the requests of all SMRs divided by 4 (the number of SMRs).The results show that our method can reduce the seek time by about 21.97% on average.The response time of Mix3, Mix7 is also lower than that of Baseline due to the increase of seek

time because of the high percentage of random data in prn_1.

5.2.3 Long Tail Latency

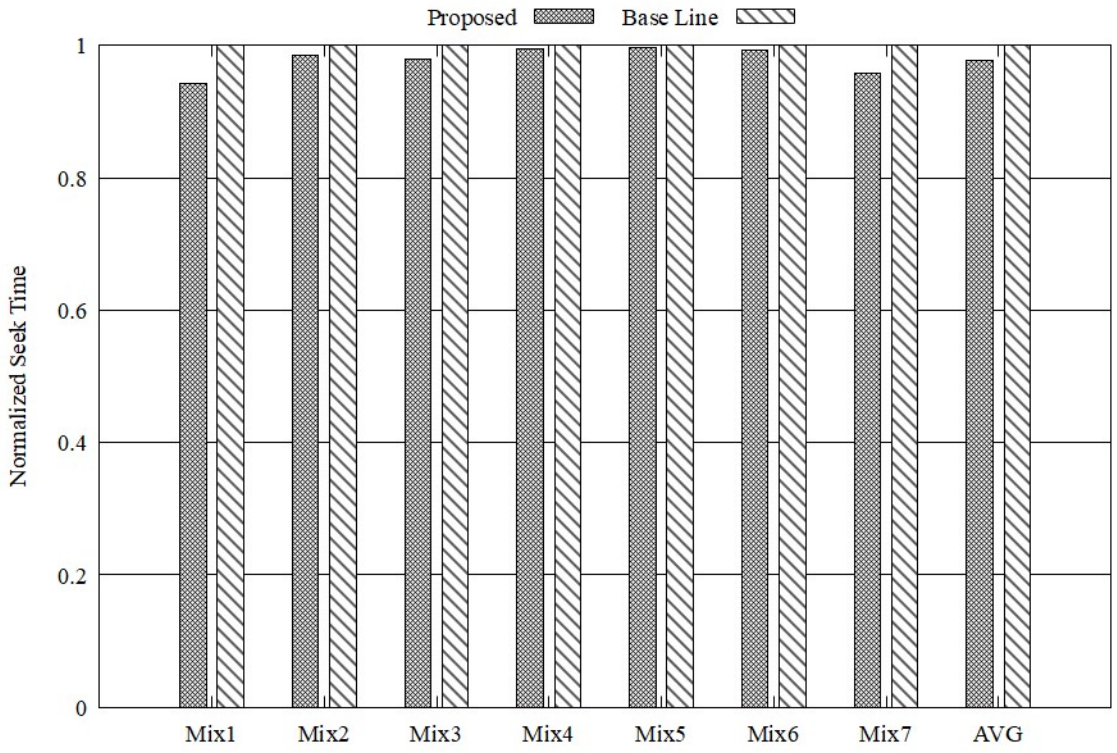

Figure 5.3: Average 99th Long Tail Lantency

As shown as Fig. 5.3 the comparison of average 99th long tail latency. The result shows that the average 99th long tail latency is reduced by about 2.27% compared with the Baseline. This is due to the fact that our proposed method classifies random data and sequential data and batch processes them separately, so it reduces the access time of a single I/O, which in turn reduces the long tail latency. This reduces the access time of a

single I/O, which in turn reduces the long tail latency.

Chapter 6 Conclusion

This study addresses the performance challenges faced by tile-based magnetic recording (SMR) hard disks during data access, and proposes a new approach that combines a hybrid access architecture, dynamic I/O classification, and revised C-LOOK scheduling method. This method fully utilizes the high-capacity advantage of SMR disks and the in-place updating feature of CMRs, and achieves a significant improvement in system performance through parallel processing of multiple hard disks.

Compared with the baseline method, the proposed method in this study reduces the average head travel time by about 133.49% and the response time by about 21.97%. The average 99th long tail latency is also significantly reduced by about 2.27% on average. This is mainly due to the effective classification and batch processing strategy for random and sequential I/O requests.

Future research directions can focus on extending the method to more diverse storage architectures and application scenarios, such as further optimizing the scheduling strategy for different types of workloads.

Reference

[1] A. Aghayev, M. Shafaei, and P. Desnoyers, “Skylight—a window on shingled disk operation,” ACM Transactions on Storage, vol. 11, pp. 16:1–16:28, October 2015.

[2] S. Zhou, E. Xu, H. Wu, Y. Du, J. Cui, W. Fu, C. Liu, Y. Wang, W. Wang, S. Sun, X. Wang, B. Feng, B. Zhu, X. Tong, W. Kong, L. Liu, Z. Wu, J. Wu, Q. Luo, and J. Wu, “Smrstore: A storage engine for cloud object storage on hm-smr drives,” pp. 395–407, 2023.

[3] T.-Y. Lin and T.-Y. Chen, “Hsmr-raid: Enabling a low overhead raid-5 system over a host-managed shingled magnetic recording disk array,” in Proceedings of the ACM SIGAPP Symposium on Applied Computing (SAC), (New York, NY, USA), pp. 294–297, ACM, 2023.

[4] Y. Pan, Z. Jia, Z. Shen, B. Li, W. Chang, and Z. Shao, “Reinforcement learning-assisted cache cleaning to mitigate long-tail latency in dm-smr,” in 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 103–108, IEEE, 2021.

[5] M. Shafaei, M. H. Hajkazemi, P. Desnoyers, and A. Aghayev, “Modeling drive-managed smr performance,” ACM Transactions on Storage, vol. 13, pp. Article 38, 22 pages, December 2017.

[6] S. Greaves, Y. Kanai, and H. Muraoka, "Shingled recording for 2–3 tb/in^2," IEEE Transactions on Magnetics, vol. 45, no. 10, pp. 3823–3829, 2009.

[7] S. Piramanayagam, "Perpendicular recording media for hard disk drives," Journal of Applied Physics, vol. 102, no. 1, pp. 1–7, 2007.

[8] G. Zhu, S. J. Lee, and Y. Son, "An efficient log-structured scheme for disk arrays," in Proceedings of the ACM SIGAPP Symposium on Applied Computing (SAC), pp. 1197–1204, ACM, 2022.

[9] L. Ma and L. Xu, "Hmss: a high performance host-managed shingled storage system based on awareness of smr on block layer," in 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), pp. 570–577, IEEE, 2016.

[10] J.-H. Chang, T.-Y. Chang, Y.-C. Shih, and T.-Y. Chen, "Lady: Enabling locality-aware deduplication technology on shingled magnetic recording drives," ACM Transactions on Embedded Computing Systems, vol. 22, pp. Article 127, 25 pages, September 2023.

[11] P. Xu, J. Wan, P. Huang, B. Shu, C. Tang, and C. Xie, "An active method to miti-

gate the long latencies for host-aware shingle magnetic recording drives," in 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), pp. 17–25, IEEE, 2019.

[12] S. Zheng, M. Hoseinzadeh, S. Swanson, and L. Huang, "Tpfs: A high-performance tiered file system for persistent memories and disks," ACM Transactions on Storage, vol. 19, pp. Article 20, 28 pages, March 2023.

[13] C. Ma, Y. Wang, Z. Shen, and Z. Shao, "Kfr: Optimal cache management with k-framed reclamation for drive-managed smr disks," in 2020 57th ACM/IEEE Design Automation Conference (DAC), pp. 1–6, IEEE, 2020.

[14] C. Ma, Z. Zhou, Y. Wang, Y. Wang, and R. Mao, "Mu-rmw: Minimizing unnecessary rmw operations in the embedded flash with smr disk," in 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 490–495, EDAA, 2022.

[15] M. H. Hajkazemi, M. Abdi, and P. Desnoyers, "μCache: a mutable cache for smr translation layer," in 2022 40th International Symposium on Computer Architecture (ISCA), pp. 1–12, IEEE, 2022.

[16] C. Ma, Z. Shen, J. Wang, Y. Wang, R. Chen, Y. Guan, and Z. Shao, "Tiler: An autonomous region-based scheme for smr storage," IEEE Transactions on Computers,

vol. 70, no. 2, pp. 291–302, 2021.

[17] B. L. Fenggang Wu and D. H. C. Du, “Fluidsmr: Adaptive management for hybrid smr drives,” ACM Transactions on Storage, vol. 17, pp. 32:1–32:30, October 2021.

[18] Q. Le, A. Amer, and J. Holliday, “Raid 4smr: Raid array with shingled magnetic recording disk for mass storage systems,” Journal of Computer Science and Technology, vol. 34, no. 4, pp. 854–868, 2019.

[19] M. A. Bamboat, M. A. Khuhro, K. Kumar, I. A. Halepoto, N. Mirbahar, and G. S. Khan, “Analysis of traditional hard disk scheduling algorithms: A review,” Quest Research Journal, vol. 19, pp. 51–58, January–June 2021.

[20] J. R. Celis, D. Gonzales, E. Lagda, and L. R. Jr., “A comprehensive review for disk scheduling algorithms,” International Journal of Computer Science Issues (IJCSI), vol. 11, pp. 74–79, January 2014.

[21] A. Shankar, A. Ravat, and A. K. Pandey, “Comparative study of disk scheduling algorithms and proposal of a new algorithm for better efficiency,” in 2nd International Conference on Advanced Computing and Software Engineering (ICACSE-2019), pp. 1–6, ICACSE, 2019.

[22] M. K. Mishra, “An improved fcfs (ifcfs) disk scheduling algorithm,” International Journal of Computer Applications, vol. 47, pp. 20–24, June 2012.

[23] S. Saha, M. N. Akhter, and M. A. Kashem, “A new heuristic disk scheduling algorithm,” International Journal of Scientific & Technology Research, vol. 2, pp. 49–54, January 2013.

[24] S. Negi, K. Singh, and S. Sahu, “Demand based disk scheduling using circular look algorithm,” International Journal of Computer Science and Mobile Computing (IJCSMC), vol. 4, pp. 364–368, August 2015.

[25] J. R. Celis, D. Gonzales, E. Lagda, and L. R. Jr., “A comprehensive review for disk scheduling algorithms,” International Journal of Computer Science Issues (IJCSI), vol. 11, pp. 74–79, January 2014.

[26] A. Thomasian, “Survey and analysis of disk scheduling methods,” ACM SIGARCH Computer Architecture News, vol. 39, pp. 8–12, May 2011.

[27] D. Narayanan, A. Donnelly, and A. Rowstron, “Write Off-Loading: Practical power management for enterprise storage,” in 6th USENIX Conference on File and Storage Technologies (FAST 08), (San Jose, CA), USENIX Association, February 2008.