AN EXT4-BASED FILE SYSTEM FOR SHINGLED MAGNETIC RECORDING DISKS

Yu-Chao Chen,
Tamkang University, Taiwan
James831202@gms.tku.edu.tw

Hsin-Wen Wei,
Tamkang University, Taiwan
hwwei@mail.tku.edu.tw

Wei-Tsong Lee
Tamkang University, Taiwan
wtlee@mail.tku.edu.tw

ABSTRACT

In the era of big data, the amount of data has risen sharply and is expected to continue to grow at a doubling rate, which forced the service providers to face the fact that capacity of traditional hard disks is insufficient to store such large amount of data generated by their clients. However, the Perpendicular Magnetic Recording technology currently used on hard disks is about to reach its storage density limit. Therefore, a new technology Shingled Magnetic Recording disk is introduced to address capacity constraints and is one of the most promising new hard disk technologies. This paper proposed an ext4-based file system for Shingled Magnetic Recording Disks to address the write amplification problem that introduced by Shingled Magnetic Disk. Through the discussion of this paper, it shows that the write amplification problem can be solved by the proposed file system.

Keywords: big data, Shingled Magnetic Recording, PMR

---

1. INTRODUCTION

The hard disk has always played an indispensable role in computer products. The shingled magnetic recording hard disk has the potential to be the main storage medium for the future information age. The Shingled Magnetic Recording (SMR) hard disk has more storage capacity than the conventional hard disk. Use overlapping tracks to increase the magnetic recording density, thereby increasing the hard disk capacity. Moreover, SMR disks adopt the disk head in the conventional disks, which has the features that the magnetic field strength required for read head of the conventional hard disk is smaller than that of the magnetic field required for write head. Therefore, the manufactures do not have to increase the cost to develop a new magnetic head when producing a SMR hard disk.

However, the disadvantage is that there are restrictions on random writing, because the some tracks are overlapped to increase the density. Therefore, how to overcome the writing limitation of the SMR hard disk and its subsequent problems

will be the focus of this paper.

In this paper, we proposed an ext4-based file system for Shingled Magnetic Recording Disks to address the write amplification problem. We propose three mechanisms to reduce the probability of random write operations. Through the discussion of this paper, it shows that the write amplification problem can be solved by the proposed file system.

2. RELATED WORK

2.1 Shingled Magnetic Recording

The number of tracks per inch that a conventional hard disk can accommodate is determined by the width of the write head, and the shingled magnetic recording technology utilizes the magnetic field strength required for hard disk reading to be lower than the strength of writing magnetic field, that is, the width of the track required for reading is smaller than the width required for writing, and the width of the read head is smaller than the width of the write head. The data track for writing operation of the shingled hard disk is partially overlapping the previous magnetic track, and at the same time, sufficient space is reserved for the narrow read head to read the data of the previous magnetic track. Since the magnetic tracks are partially overlapped, the write operation will cover several read tracks, as shown in Figure 1. Therefore, the random write operation will introduce write amplification problem, write a track will destroy the data in several subsequent tracks. Sequential write operation is preferred by SMR disks. To address the issues in SMR disks, the solutions are further classified into three classes as described in next subsection.

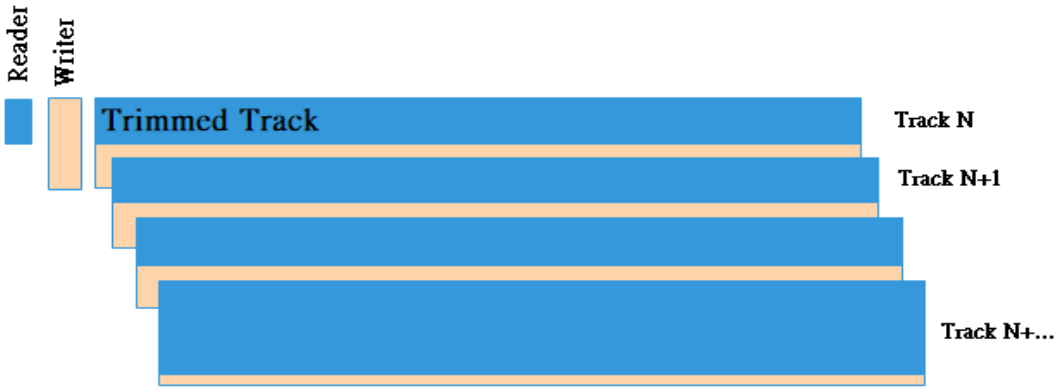

Figure 1. Overlapping Tracks in SMR Technology

2.2 SMR Disk Layout and definition

The shingled hard disk divides the track into several bands, that is, a continuously writeable area composed of continuous tracks, each of which constitutes a basic unit requiring sequential writing, as shown in Figure 2. Band is the physical concept of the SMR hard disk, and its corresponding logical concept is the zone, which is the sequential writing area that the upper layer software can see. Since the SMR can still ensure that data is read from the non-overlapping portions of the track, data reading within the Zone can still be done at random. The SMR hard disk is divided into partitions at the time of manufacture, and the starting position of the zone is identified by leaving a large interval between the zones. In addition, some of the traditional hard disk tracks are preserved in a small area of the SMR hard disk, called the

Conventional Zone which provides a foothold for the random write operation of the upper application. Other zones are detailed as below.

Figure 2. SMR Band Structure

- **Sequential-Write-Priority Zone**

Each Sequential Write Priority Zone has a Write Pointer, which indicates the location of write in the zone area. The application should write at the write pointer first, it also can handle random writes with some penalty, since random write will lead to subsequent garbage collection issues and so on.

- **Sequential-Write-Required Zone**

It is possible to know the position of the write pointer, but only accepts the write data at the position of the current pointer and sequentially appends the write, and refuses to write the rest of the LBA except the write pointer.

According to the above design and definition, we can further classify the shingled hard disk into three different modes: Drive Managed, Host Managed, Host Aware.

2.2.1 Drive Management SMR

The shingled magnetic recording is used, but the Zone and Write Pointer are hidden from the host, and the order of the write is managed by the firmware, which is called Drive Managed SMR. Using the same interface as the traditional hard disk, a shingled translation layer (STL) is implemented internally to hide the sequential write limitation of the SMR hard disk from the host, so the software does not need to be changed. However, under some workloads with random writes, the sequential write area inside the DM-SMR requires background operations such as data migration and garbage collection. Performance may be unpredictable and can be affected by IO patterns. Weiping He and David HC Du proposed SMaRT [1] to optimize the performance of DM-SMR and use the Singled Translation Layer to convert random writes into sequential writes. DM-SMR devices can be used in traditional system for good compatibility with existing hard disk systems.

2.2.2 Host Management SMR

With shingled magnetic recording, the host is provided with a Zone and Write Pointer according to the ZBC/ZAC standard, and only the Conventional Zone and the Sequential-Write-Required Zone are included in, which is Host-Management SMR (HM-SMR). HM-SMR provides a new interface for above software and application and is not backward compatible to previous system. HM-SMR allows the upper layer software to manage the sequential write limit of the hard disk, and the IO request that

does not match the sequential write limit will be refused to write, because the IO of the hard disk is completely controlled by the software, so its performance is predictable and can represent the biggest advantage of SMR hard disk. Liuying Ma and Lu Xu proposed HMSS [2] to make partial modifications to the mechanism of host management SMR.

2.2.3 Host Aware SMR

With shingled magnetic recording, the host is provided with a Zone and Write Pointer according to the ZBC/ZAC standard, and the Conventional Zone and the Sequential-Write-Priority Zone are included in, which is called Host Aware SMR (HA-SMR). HA-SMR is a compromise model that combines the features of drive management and host management. It provides a new interface like host management, but unlike host management, it is backward compatible and when it receives non-conforming sequences, the host senses that SMR will write the data instead of rejecting it. XUCHAO XIE et al. proposed SMRC [3], which combined the SSD and HDD to optimize the write performance of HA-SMR system.

This paper will mainly optimize the performance of the host-aware shingled hard disk, and introduce the improvement methods and mechanisms in the following section.

3. METHODOLOGY

3.1 Ext4 file system

Fourth extended file system (EXT4) is a file system widely used in Linux and is the first file system designed specifically for Linux. The disk layout of EXT4 is shown in Figure 3. First, EXT4 will divide the hard disk space into many groups, and each group is 128MB. The Group contains Super block, Group Descriptor block, Block Bitmap Block, and Inode Bitmap Block, Inode Table, and Data Blocks.

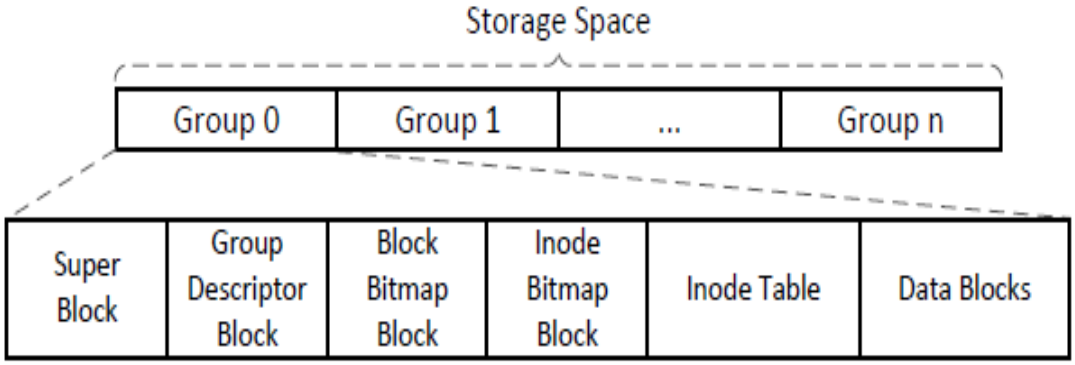

Figure 3. EXT4 file system structure

Super Block: used to record the overall information of the file system, such as: file system size, available blocks, number of inodes, and unused Inode lists...etc.

Group Descriptor block: mainly used to record information about the group, such as the number of blocks available for the group, the number of available inodes, and the location of the Block Bitmap Block, the Inode Bitmap Block, and so on.

Block Bitmap Block: used to record the usage status of Data Blocks, so that the system can know whether each block in the Group's Data Blocks has stored data.

Inode Bitmap Block: used to record the usage status of the Inode Table. The

system knows whether the Inode in the Group's Inode Table is used or not used.

Inode Table: used to store all the Inodes in this group, and find the required files and the blocks of the stored data through the Inode Table.

Data Blocks: used to store data.

3.1.1 Inode

The EXT4 file system uses the inode data structure, as shown in Figure 4, to record the metadata of the file. A file will occupy an inode and give an inode number. The metadata contained in the inode has the size of the file, the permissions, and the location where the data is stored...etc. Basically, all information except the file name is included. When a file is requested, the file system will divide the action into three processes. First, find the inode number corresponding to the file, then find the inode data through the inode number. After obtaining the inode data block, the storage block used by the data can be found through the information in the inode. Finally, the data can be read from the disk. With file's inode data, many attributes, such as access time, storage blocks, modification time, can be obtained.

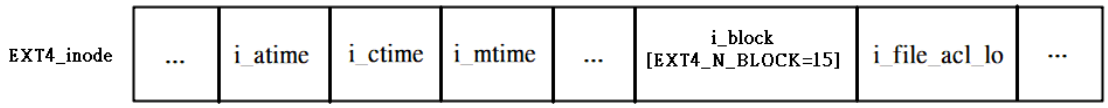

Figure 4. EXT4_inode structure

i_atime：Last access time, in seconds since the epoch.
i_ctime：Last inode change time, in seconds since the epoch.
i_mtime：Last data modification time, in seconds since the epoch.
i_block[EXT4_N_BLOCKS=15]：Block map or extent tree.
i_file_acl_lo：Lower 32-bits of extended attribute block.

However, we still need more information about the requested files, such as popularity and modification frequency. Hence, we use "Extended Attributes" structure in Ext4 to record the access counts of the requested data. Extended attributes[4] structure is a feature of Ext4 file system, which allows the user to associate computer files with metadata that is not used by the file system, and as an extension to record more information about the data for access control, security, or time precision issues. Extended attributes are usually stored in separate data blocks on the hard drive and referenced from the inode via inode.i_file_acl. The beginning of an extended attribute block is in struct ext4_xattr_header, which is 32 bytes long, as shown in Figure 5. With the extended attribute block we can further record some information needed for the system to rearrange the stored location of data.

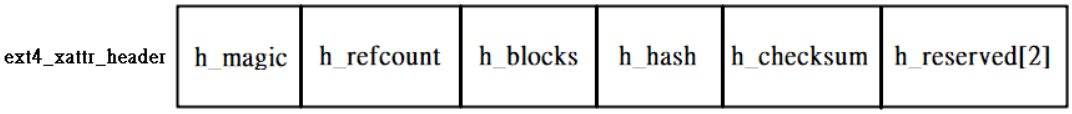

Figure 5. ext4_xattr_header structure

h_magic：Magic number for identification.
h_refcount：Reference count.

h_blocks：Number of disk blocks used.
h_hash：Hash value of all attributes.
h_checksum：Checksum of the extended attribute block.
h_reserved[2]：empty.

3.2 Proposed mechanisms

Based on the EXT4 Inode structure, we made some modifications to the Extended Attributes in the inode structure. We replace h_refcount field and use this field to the number of times that the file is modified for a certain period of time, and re-defined the field as h_time change and replace the h_reserved field to mark the file as cold data or hot data and re-define it as h_mark. The modified structure is shown in Figure 6.

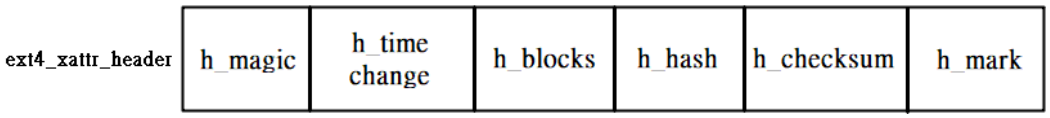

Figure 6. Modified ext4_xattr_header structure

With the modified structure, we then propose three mechanisms to reduce the write overhead of SMR hard disk: file classification, dual-mode file update, and space reorganization, and gradually describe each mechanism as below.

- **File classification**

First, we classify the data files which are to be written to disks into cold data files and hot data files, and determine the track position of the data to be written. Different from the general definition of cold data and hot data; the so-called cold data and hot data in this paper, the former refers to files that are not likely to be changed or modified, such as: photos or videos etc. The latter refers to files with frequent changes or revision, such as: txt or .doc text files etc. Since the hot data may be changed frequently, the system intends to write hot data to the outer track, while the part of the cold data is written to the inner track.

In addition, we have defined the inner and outer rings of the magnetic track separately. Since the reading rate of the outer ring is larger than the reading rate of the inner ring, and it is more suitable for storing hot data with high read/write rate.

- **Dual-mode file update**

In terms of the frequency of data writing, file update mode is mainly divided into two types: in-place-write and out-of-place-write. The so-called in-place-write refers to the first write of the data when the data is written to the file system, when the data is to be updated; the indexed data storage blocks are updated. Out-place-write refers to the general write mode. When data is written to the file system, the data is written to the free blocks for the first time, however, when the data is updated, the updated data is written in the new blocks instead of the old blocks.

We write cold data to the inner track of the disk and use in-place-write to make changes to the data when updates are needed, the reason for taking the in-place write is because the cold data is usually a larger data and likely occupy all tracks in one band. Hence, it would not likely introduce write amplification problem. On the other hand, if the data update of the storage block is frequently performed, the write delayed of the data update will be extended, since update the old data block, in order not to affect the

data on the previous track, must rewrite the data of the entire tracks in the same band, which will cause the write amplification issue of the hard disk.

Therefore, for the hot data, the out-place-write method is applied. The main reason for taking the out-place-write is to conform to the sequential write principle of the host-managed shingled hard disk, and write the data into the new block instead of updating the block of the original old data. In addition, whenever the file is updated, because h_mark will mark the data as cold data or hot data, the file system can know the data attribute. Note that, we use h_time to record the number of changes in the data during the specific period of time.

- **Space Reorganization**

Because the out-place-write method is applied to update the file, it is necessary to have a space recycling mechanism to reclaim the old space that is no longer needed to fulfill the request of subsequent data writing. First, when the remaining space capacity of the hard disk is under 20%, we will start the spatial reorganization, release the old data block, and move the data according to the access frequency of the data.

Since we first treat the first written data as hot data and store it in the block of outer track, we give 3 days as a period to determine whether the data is cold or hot. If the new data has not been modified within 3 days, we will treat it as cold data, and the opposite is hot data. In addition, if the hot data has not been modified in 7 days, it will become cold data, and if the cold data has a modification action during this period, it becomes hot data.

Then, the time to wait for the data is moved from the longest to the shortest to the released spatial block. Assuming that the space of the outer ring track is full, it is preferred to move the data with a long access time to the inner track to enable new data to be written. If the storage space of the inner ring track is full, the middle track is selected as the moving position, and the data with the shorter inner track access time is moved. In addition, we will also set a background time for the hard disk to do regular spatial reorganization. The space for hard disk is increased by the mechanism of spatial reforming. Figure 7 shows the flow of data written to the hard disk.

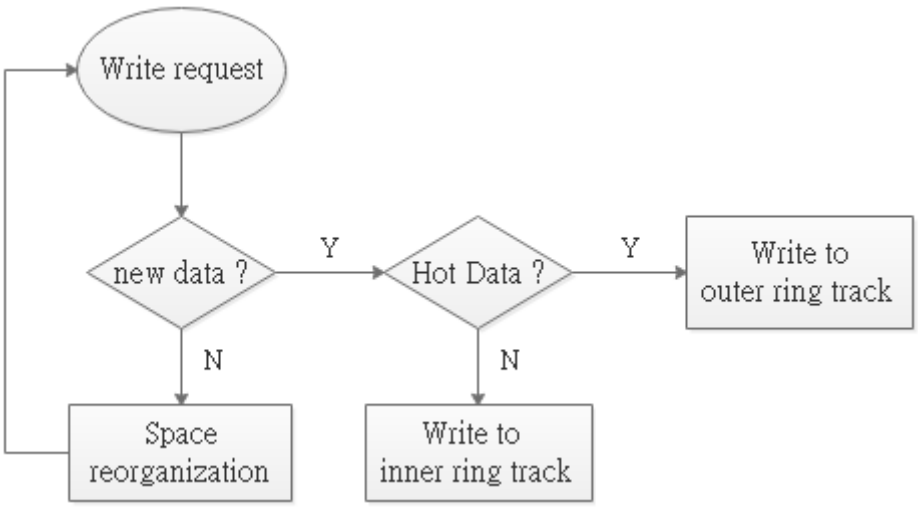

Figure 7. flowchart of write/update request

4. DISCUSSION

In this section, we give an example to illustrate the mechanism structure and action principle proposed in this paper. Generally, when a write request is arrived, there will

be so-called difference between hot and cold data, but this is the information that the file system will not know, so we first regard the data written for the first time as hot data, and h_mark will first mark the data as hot data. Since the write direction of the hard disk is written from the inside to the outside, according to our definition, the cold data is written to the inner track (IT), and the hot data is written to the outer track (OT). The dotted square is a block that holds the data track, as shown in Figure 8.

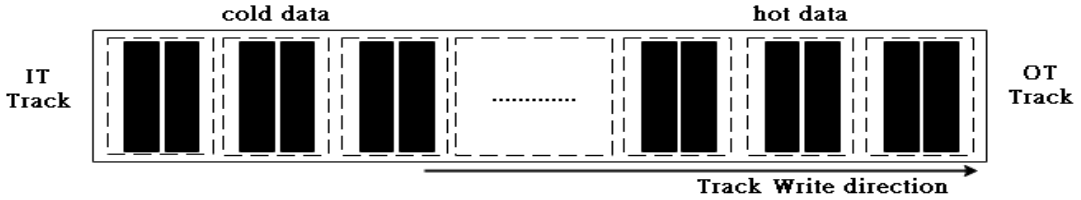

Figure 8. Track position of the data write

In terms of file writing, there are in-place writes and out-of-place writes. Due to the frequent update of the hot data, we use the out-of-place write method and write the updated data in the current pointer position according to the sequential write limit of the shingled hard disk, as shown in Figure 9(a). And then the old space is released for use in space reorganization stage. The part of the cold data is written in-place. If the cold data becomes hot and is to be moved, the inode of the data storage area will be updated by changing the indicator, as shown in Figure 9(b), and let the file system know the updated location.

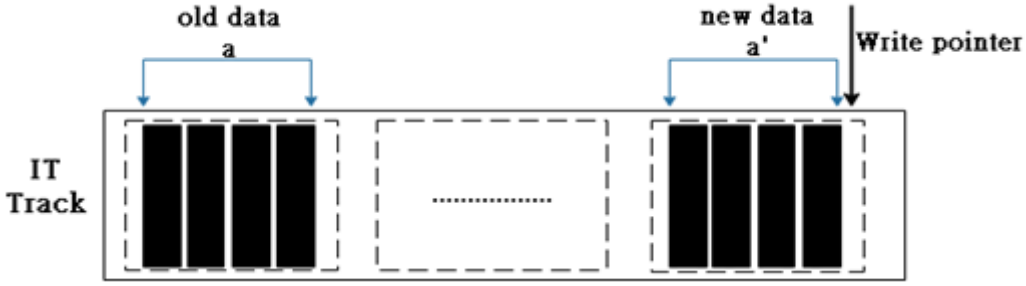

Figure 9(a). out-of-place write

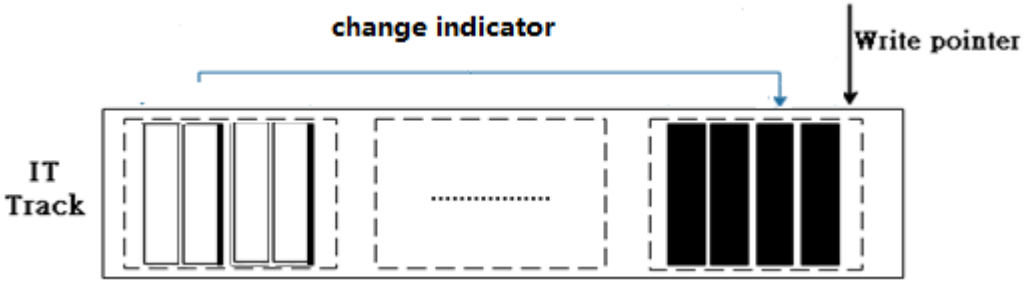

Figure 9(b). change indicator

When the remaining capacity of the hard disk is under 20%, we will start spatial re-construction to reclaim the old space and release the new space for subsequent data writing. The file information recorded by h_time change and h_mark is used to determine the hot and cold condition of the data and the order of data movement. Since

the data is first marked with hot data by h_mark, according to the mechanism of spatial reforming, we will first make some judgments on the data to determine whether it is still hot data or has been converted into cold data.

Through the file information recorded by h_time change, the file writing time, modification time, etc. can be obtained. When the space reorganization is started for the first time, it will first determine whether the first written data has been modified within 3 days. If the file has to make changes within 3 days, then regarded as hot data, and vice versa is cold data. During the subsequent reorganization, if the judged data is not written for the first time, it is determined by 7 days whether the data has changed during this period. If the original data is hot data but did not change in the 7 days prior to the period after the second space reforming, then turned it into cold data, and vice versa remains in hot data. And h_mark will update the hot and cold tags of the data for the next round. By doing this, the write amplification overhead can be reduced.

In addition, when the outer space of the track is full, space reforming will be initiated to release the old space and perform space movement to write subsequent data. If the inner track is also full, the middle track (MT) is selected as the moving position, and the data with the shorter inner track access time is selected as the moving object, because we think the data may have just changed from hot to Cold, as shown in Figure 10.

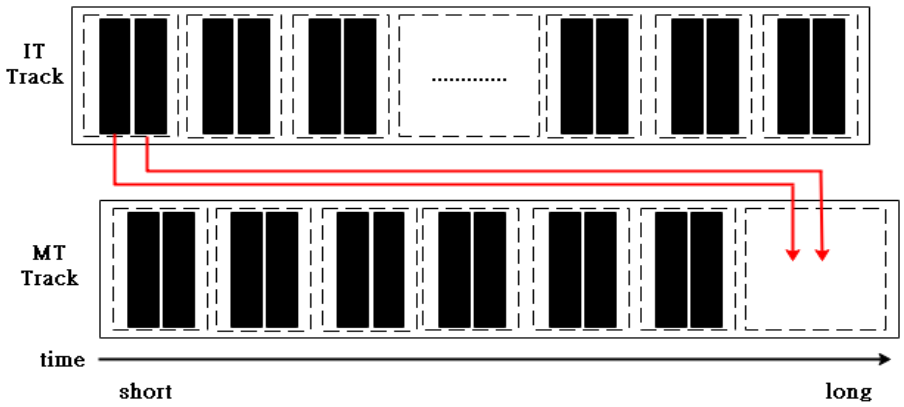

Figure 10. Data movement of the inner track of the track

5 CONCLUSION AND FUTURE WORK

Through the method mechanism of this paper, a suitable writing method is provided for the write limitation of the shingled hard disk. First, we will classify the different data into hot and cold data, which is judged by the update frequency of the data. Then update the file by means of in-place and out-of-place writing to reduce the write amplification and read/write speed of the hard disk. Space reorganization can reclaim old space and allow subsequent data to be continuously written to maximize the use of hard disk capacity.

For future work, we will conduct the experiments on shingled hard drives and traditional hard drives, and then compare the results of our proposed method to that of the original Extr4 file system.

6 REFERENCES

[1] Weiping He, David H.C. Du. “SMaRT: An Approach to Shingled Magnetic Recording Translation”, 15th USENIX Conference on File and Storage Technologies (FAST ’17). February 27–March 2, 2017

[2] Liuying Ma, Lu Xu. “HMSS: A High Performance Host-Managed Shingled Storage System Based on Awareness of SMR on Block Layer”, 2016 IEEE 18th International Conference on High Performance Computing and Communications
[3] XUCHAO XIE, LIQUAN XIAO, XIONGZI GE, QIONG LI. ” SMRC: An Endurable SSD Cache for Host-Aware Shingled Magnetic Recording Drives”, IEEE Access, Date of Publication: 09 April 2018
[4] EXT4 Retrieved https://ext4.wiki.kernel.org