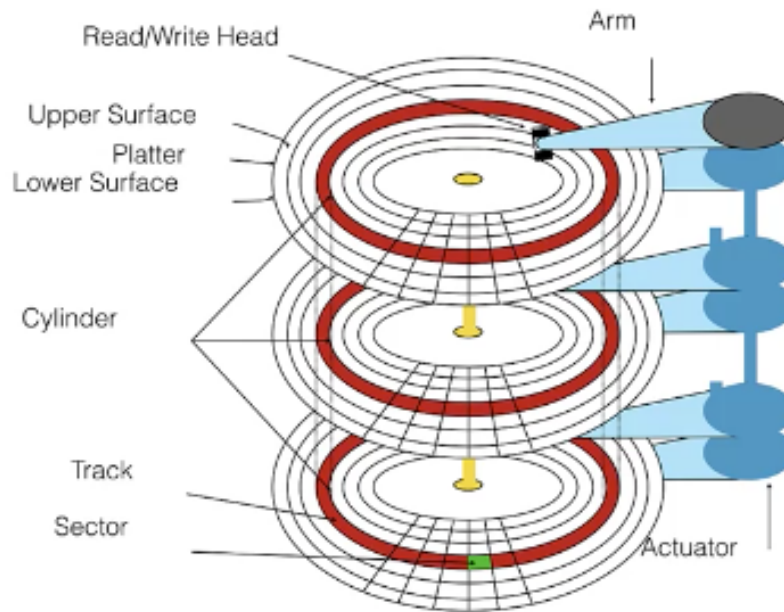


1. Secondary Storage Devices

- Focus will be on hard-drives

2. Disk Components



- Parts
 - **Platter:**
 - * Data can be stored in both upper and lower parts of the platter
 - **Cylinder:**
 - * Is a set of tracks that can be read without moving the arm
 - **Sector:**
 - * Size of disk block is multiple of sectors
- Disk surface crash



- Occurs when disk arm touching surface

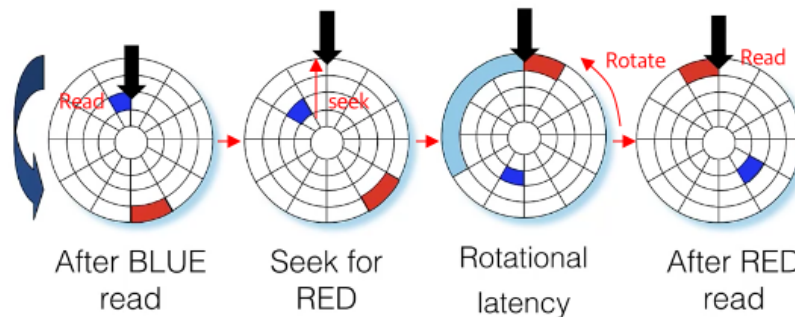
- Results in permanent loss of information on the track

3. Disk Performance

IMPORTANT We should know the bulk part time of how this works

- **Seek:**
 - Is the time it takes to move the disk arm to correct cylinder
 - Depends on how fast disk arm can move
 - Typical time: 1-15ms, depending on distance (avg 5-6 ms)
 - Improves very slowly (7 - 10% per year)
- **Rotation:**
 - Is the time it takes to rotate under the head to get to correct sector
 - Depends on rotation rate of disk
 - Average latency of $\frac{1}{2}$ rotation
- **Transfer:**
 - Is the time it takes to transfer data from surface to disk controller, electronics and sending it back to host
 - Depends on density
 - $\sim 100\text{MB/s}$, average sector transfer time of $\sim 5\mu\text{s}$
 - Improves rapidly ($\sim 40\%$ per year)

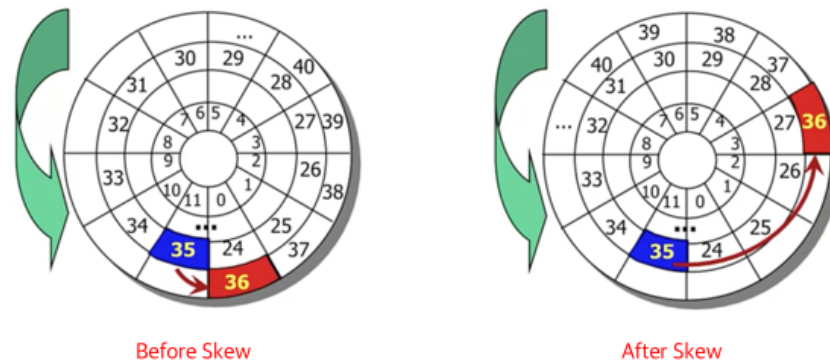
4. Traditional Service Time Component



- OS tries to minimize the cost of rotational latency, transfer time, and seek time
- Improvement attention especially on seek time and rotation latency

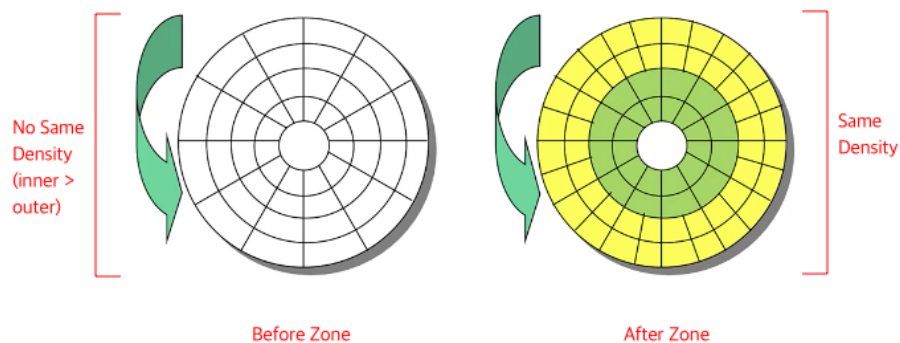
5. Some Hardware Optimizations

- **Track Skew**



- Has to do with numbering on tracks
- Is to reduce rotational latency

- **Zones**



- Is to make sure data is stored with same density
- Is done to maximize the capacity of hard drive
- Outer tracks → holds more sectors

- **Cache**

- Is also called **Track Buffer**
- Is a small memory chip embedded in hard drive (8 – 16MB)
- Is aware of disk geometry
- May cache whole track
- Boosts future reads on the same track

6. Disk and the OS

- The OS provides different levels of disk access to different clients
 - Physical disk (e.g surface, cylinder, sector)

IMPORTANT Logical disk (disk block #) ← what we will do for the first assignment

- Logical file (e.g file block, record, or byte #)

• Enhancing Disk Performance

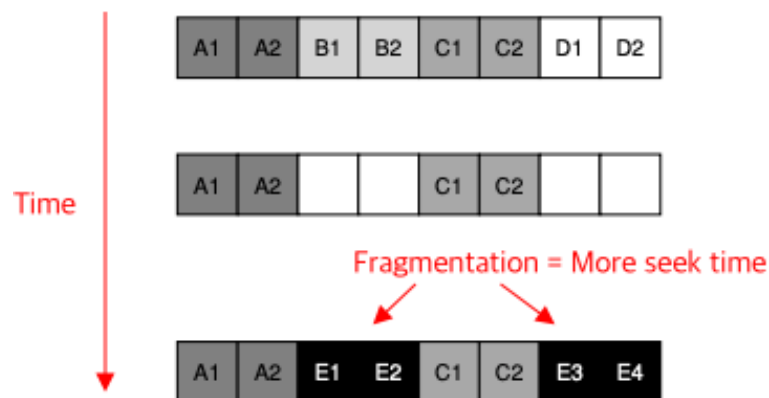
- File system needs to be aware of disk characteristics for performance
 - * **Allocation Algorithm** → enhances performance
 - e.g Extent-based allocation, indexed based allocation, linked-based allocation
 - * **Request Scheduling** → reduce seek time
 - e.g. FCFS, SSTF, SCAN, C-SCAN
- Disk characteristics yields to goals:
 - * **Amortization**
 - Compensates positioning delay
 - Grabs lots of useful data while at it
 - Performance improvement upto factor of 10
 - * **Closeness**
 - Done by putting things close to each other
 - Performance benefit in factors of 2

• Allocation Strategies

- Disk perform best if seeks are reduced and large transfers are used
 - * Done by allocating data close together
 - * Reason why significant improvement in seek time and transmission time over the years

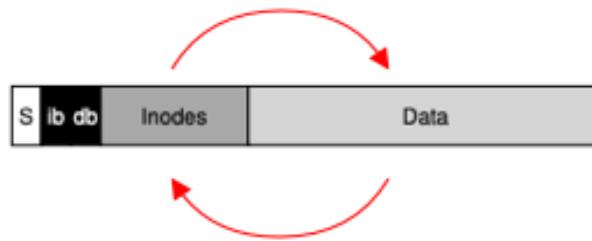
• Original Unix File System

- Is simple and straightforward
- Is slow (poor use of disk bandwidth)
- Has 2 placement problems
 1. Fragmentation



- * Causes more seeking
- 2. The travel of back and forth between inode and data blocks

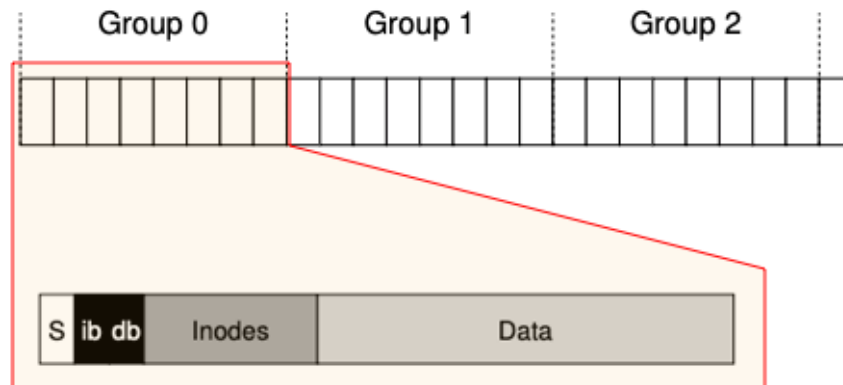
Disk arm moving back and forth = Lots of seek time



* More seeking time

• Fast File System

- Is a disk aware file system
- Addressed placement problems using **cylinder groups**



* Steps

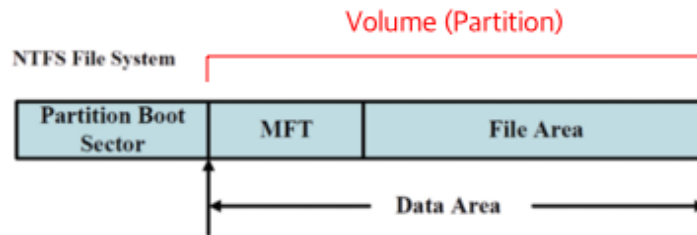
- Data blocks in the same file allocated in same cylinder group
- Files in the same directory allocated in same cylinder group
- Inodes for files allocated in same cylinder group as file data blocks

* Allocation in cylinder groups provide closeness → less long seeks

* Has **Free space requirements**

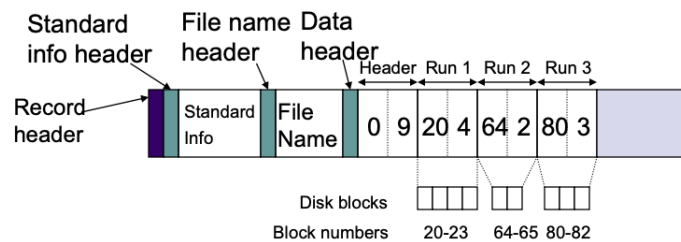
- requires free space to be scattered across disk to allocate properly using **cylinder groups**
- 10% of total disk space in each **cylinder group** is reserved for this
- Doesn't like filling up one cylinder group
- Large file is allocated by breaking into cunkhs and storing each in different cylinder groups
- Allocates near by cylinder group if preferred cylinder group is full

• NTFS



- Is replacement of old FAT file system
- Uses extent-based allocation
 - * Tries to allocate files in consecutive blocks
- Each volume is a linear sequence of blocks (usually 4KB in size)
- Each has a **master file table**
 - * Is 1KB (or Kib) long
 - * One or more records per file or directory
 - Is analogous to **inode**
 - * Long attributes can be stored externally, and a pointer kept in MFT record

• MFT Record



- Is analogous to inode
- Each data attribute indicates the starting block and the number of blocks in a run (or extent)
- If all records are large and one MTF record is not enough, extension record is used to hold more