1 Hard disks

Disks are constructed as multiple aluminium/glass platters covered with **magnetisable material**.

- **Read/write** heads fly just above the surface (0.2 0.07mm) and are connected to a single disk arm controlled by a single **actuator**
- **Data** is stored on both sides
- Common **diameters** range from 1.8 to 3.5 inches
- Hard disks **rotate** at a **constant** speed (i.e., speed on the inside less than on the outside)

A disk controller sits between the **CPU** and the **drive**. Hard disks are currently about **4** orders of magnitude slower than main memory.

1.1 Low level format

Disks are organised in:

- Cylinders: a collection of **tracks** in the **same relative position** to the *spindle*
- Tracks: a concentric circle on a single platter side
- Sectors: segments of a track (usually 512B or 4KB in size)

Sectors usually have an **equal number** of bytes in them, consisting of a **preamble, data**, and an **error correcting code**. The number of sectors **increases** from the **inner side** of the disk to the outside

1.2 Organisation

Disks usually have a **cylinder skew**: i.e., an offset is added to sector 0 in adjecent tracks to account for the seek time. In the past, consecutive disk sectors were **interleaved** to account for transfer time. Note that as a result of this low-level formatting, disk capacity is **reduced** (size of preamble, ECC, etc)

1.3 Access

- Access time = seek time + rotational delay + transfer time
- Seek time = time needed to move the arm to the cylinder (dominant)
- Rotational latency = time before the sector appears under the head (on average half the rotation time)
- Transfer time = time to transfer the data

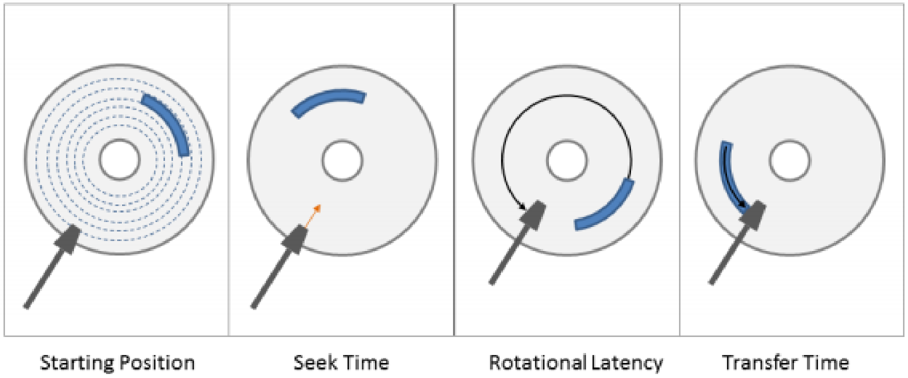

Figure: Access time to Disk (Source: www.studiodaily.com/)

Multiple requests may be happening at the same time (**concurrently**). Thus, access time may be **increased** by a queueing time. In this scenario, dominance of seek time leaves room for **optimisation** by carefully considering the **order** of read operations. The estimated *seek time* (i.e., to move the arm from one track to another) is approximated by:

$$T_s = n * m + s$$

In which :

- T_s denotes the **estimated seek time**
- n the **number of tracks to be crossed**
- m the **crossing time per track**
- s any additional **startup delay**

It is important to **position** the sectors carefully and avoid *disk fragmentation.*

2 Disk scheduling

The OS must use the hardware **efficiently**:

- The file system can **position/organise** files strategically
- Having multiple disk requests in a queue allows us to **minimise** the arm movement

Note that every I/O operation goes through a **system call**, allowing the operating system to **intercept** the request and resequence it. If the drive (or the controller) is free, the request can be serviced **immediately**, if not, the request will be **queued**.
In a **dynamic** situation, several I/O requests will be made over time that are **kept** in a table of requested sectors per cylinder. Disk scheduling algorithms **determine** the order in which disk events are processed.

2.0.1 Algorithms

- First come first served (FCFS)
  - perform operations in order requested
  - no reordering of work queue
  - no starvation: every request is serviced
  - poor performance
- SSTF (Shortest Seek Time First)
  - after a request, go to the closest request in the work queue, regardless of direction
  - reduces total seek time compared to FCFS
  - Disadvantages
    - starvation is possible; stay in one area of the disk if very busy
    - switching directions slows things down
- SCAN
  - keep moving in the same direction until end is reached (start upwards)
    - It continues in the current direction, servicing all pending requests as it passes over them
    - When it gets to the last cylinder, it reverses direction and services all the pending requests (until it reaches the first cylinder)
  - (Dis-)advantages include:
    - The upper limit on the waiting time is 2 number of cylinders, i.e. no starvation occurs
    - The middle cylinders are favoured if the disk is heavily used (max. wait time is N tracks, 2N for the cylinders on the edge)
- LOOK

    - like SCAN but stops moving inwards (or outwards) when no more requests in that direction exist.

- C-SCAN (circular scan)
    - The disk arm moves in one direction servicing requests
    - When it gets to the last cylinder of the disk, it reverses direction but it does not service requests on the return journey
    - Once it gets back to the first cylinder it reverses direction, and again services requests
    - It is **fairer** and **equalises** response times across a disk

- C-LOOK
    - moves inwards servicing requests until there are no more requests in that direction, then it jumps to the outermost outstanding requests.
    - repeast this over and over.
    - variant: service requests from inside to outside, then skip back to the innermost request.

3 Driver caching

For most current drives, the time required to seek a new cylinder is **more** than the rotational time (remember pre-paging in this context!). It makes sense, therefore, to **read more** sectors than actually required. Read sectors during the rotational delay (i.e. that **accidentally pass by**). Modern controllers read **multiple sectors** when asked for the data from one sector: *track-at-a-time caching.*

4 SSD

Solid State Drives (SSDs) have no moving parts and store data using electrical circuits. They dont have `Tseek` or rotational delay!
FCFS algorithm is useful in general purposes systems
SSTF, SCAN, LOOK-SCAN may reduce performance (no heads to move)

5 Notes

Take a look at slides for algorithm time calculation examples.

Reference section

spindle

A spindle is a shaft that holds rotating hard disk drive (HDD) platters in place. The term is also often used to refer to a single HDD.