Ceng 302
Database Managment Systems

Physical Data Storage Devices

Prof. Dr. Adnan YAZICI
Department of Computer Engineering,
Middle East Technical University
(Fall 2021)

Secondary Storage Devices

Two major types of storage devices:

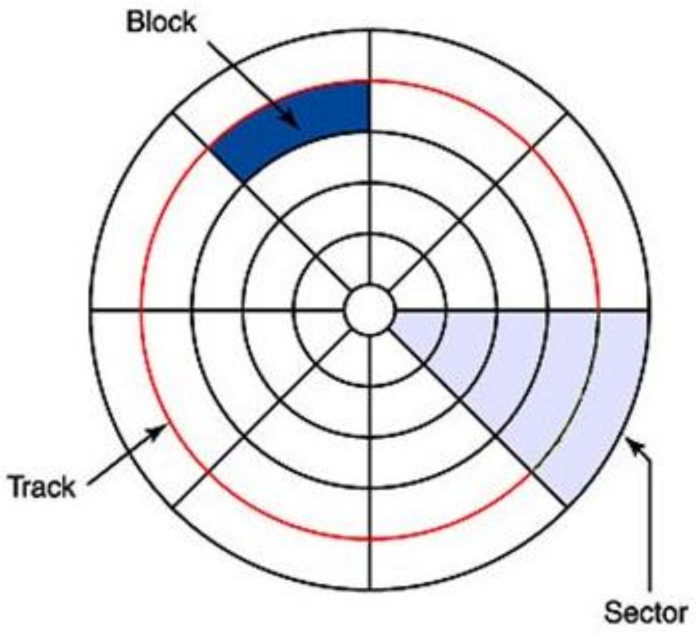
1. Direct Access Storage Devices (DASDs)
   - **Magnetic Disks**
     - Hard disks (high capacity, low cost per bit)
     - Floppy disks (low capacity, slow, cheap)
   - **Optical Disks**
     - CD-ROM = (Compact disc, read-only memory)
2. Serial Devices
   - **Magnetic tapes** (very fast sequential access)

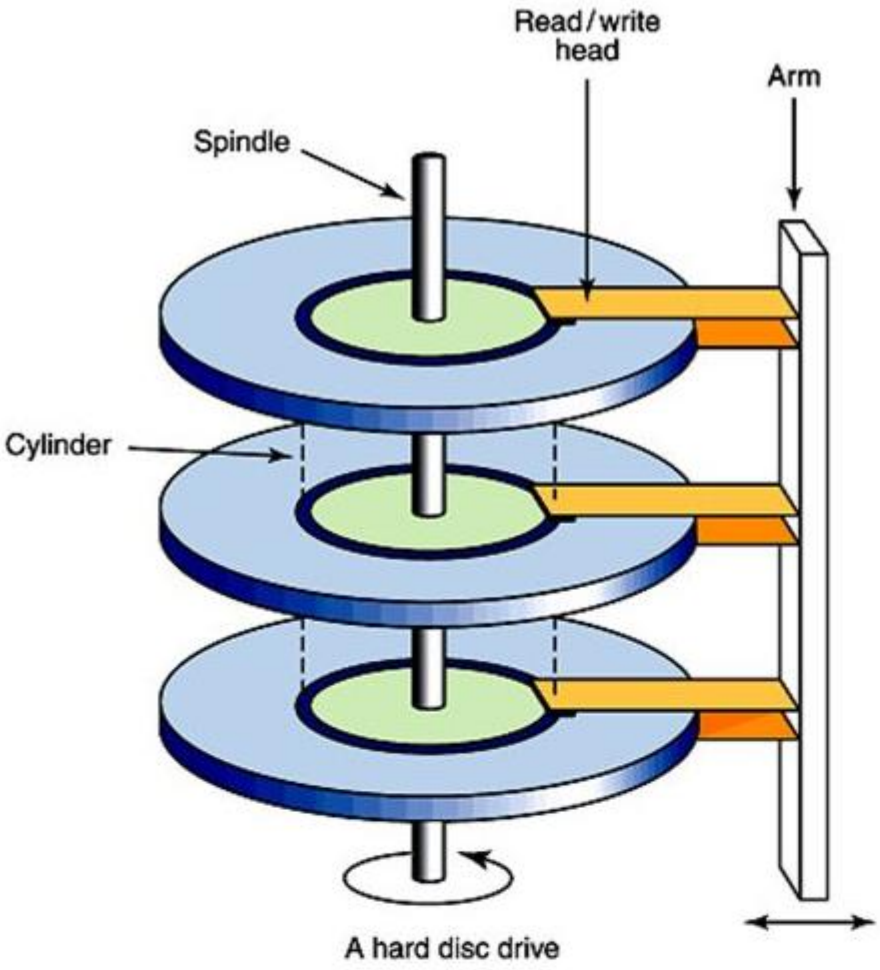
Magnetic Disks

- Bits of data (0's and 1's) are stored on circular magnetic platters called **disks**.
- A disk rotates rapidly (& never stops).
- A **disk head** reads and writes bits of data as they pass under the head.
- Often, several platters are organized into a **disk pack** (or **disk drive**).

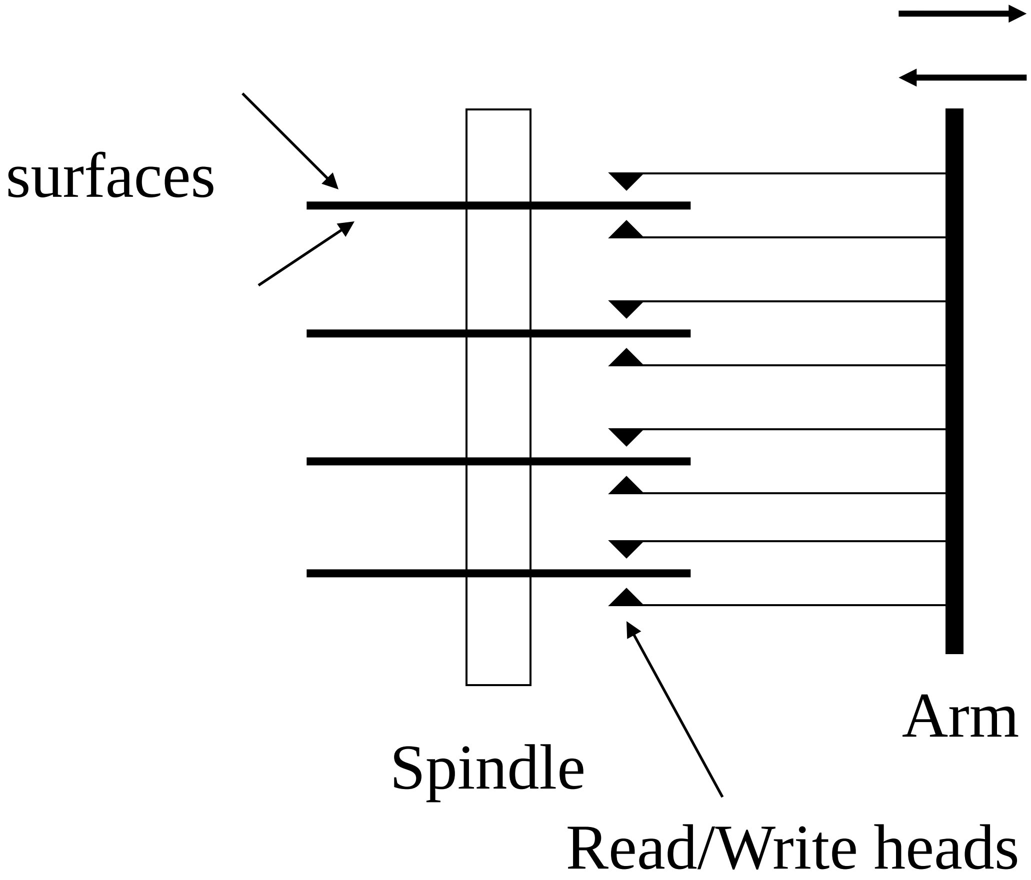
Magnetic Disks

- A read/write head travels across a spinning magnetic disk, retrieving or recording data
- Each disk surface is divided into sectors and tracks
- Example of disk addressing scheme: surface 3, sector 5, track 4





A Disk Drive



Disk drive with 4 platters and 8 surfaces

Organization of Disks

- Disk contains concentric **tracks.**
- Tracks are divided into **sectors**
- A **sector** is the smallest addressable unit in a disk.

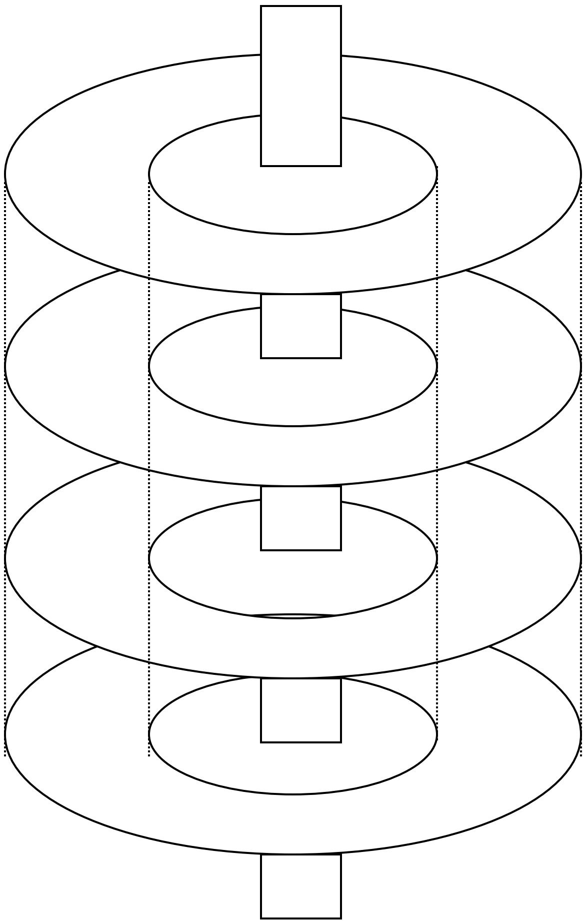
Accessing Data

- When a program reads a byte from the disk, the operating system locates the surface, track and sector containing that byte, and reads the entire sector into a special area in main memory called **buffer**.
- The bottleneck of a disk access is moving the read/write arm. So, it makes sense to store a file in tracks that are below/above each other in different surfaces, rather than in several tracks in the same surface.

Cylinders

- A **cylinder** is the set of tracks at a given radius of a disk pack.
  - i.e., a cylinder is the set of tracks that can be accessed without moving the disk arm.
- All the information on a cylinder can be accessed without moving the read/write arm.

Cylinders



Estimating Capacities

Track capacity = # of sectors/track * bytes/sector

Cylinder capacity = # of tracks/cylinder * track capacity

Drive capacity = # of cylinders * cylinder capacity

Number of cylinders = # of tracks in a surface

Exercise

- Store a file of 20000 records on a disk with the following characteristics:

  # of bytes per sector = 512

  # of sectors per track = 40

  # of tracks per cylinder = 12

  # of cylinders = 1331

Q1. How many cylinders does the file require if each data record requires 256 bytes?

Q2. What is the total capacity of the disk?

Clusters

- Another view of sector organization is the one maintained by the O.S.'s **file manager**.
- It views the file as a series of **clusters** of sectors.
- File manager uses a **file allocation table (FAT)** to map logical sectors of the file to the physical clusters.

Extents

- If there is a lot of room on a disk, it may be possible to make a file consist entirely of contiguous clusters. Then we say that the file is one **extent**. (very good for sequential processing)
- If there isn't enough contiguous space available to contain an entire file, the file is divided into two or more noncontiguous parts. Each part is an extent.

Fragmentation

Internal fragmentation: loss of space within a sector or a cluster.

1) **Due to records not fitting exactly in a sector**: e.g., Sector size is 512 and record size is 300 bytes. Either
   - store one record per sector, or
   - allow records *span* sectors.
2) **Due to the use of clusters:** If the file size is not a multiple of the cluster size, then the last cluster will be partially used.

Choice of cluster size

Some operating systems allow system administrator to choose cluster size.

- When to use large cluster size?
- What about small cluster size?
  - The greater the block-size, the greater potential amount of internal track fragmentation.
  - The flexibility introduced by the use of blocks rather than sectors can save time since it lets the programmer determine, to a large extent, how the data is to be organized physically on disk.

Organizing Tracks by Block

- **Disk tracks** may be divided into **user-defined blocks** rather than into **sectors**.
- **Blocks** can be fixed or variable length.
- A **block** is usually organized to hold an integral number of logical records.
- **Blocking Factor** = the number of records stored in a block.
- No internal fragmentation, no record spanning two blocks.
- In block-addressing scheme each block of data is accompanied by one or more *subblocks* containing extra information about the block.

Non-data Overhead

- Both blocks and sectors require non-data overhead (written during formatting)
- On sector addressable disks this information involves sector address, track address, and condition (usable/defective). Also pre-formatting involves placing gaps and synchronization marks.
- On block-organized disk, more information is needed and the programmer should be aware of some of this information.

Non-data Overhead

- Whether using a **block** or a **sector** organization, some space on the disk is taken up by non-data overhead. i.e., information stored on the disk during pre-formatting.
- On **Sector-Addressable disks**, pre-formatting involves storing, at the beginning of each sector, sector address, track address and condition (usable or defective) + gaps and synchronization marks between fields of info to help the read/write mechanism distinguish between them.
- On **Block-Organized disks**, subblock + interblock gaps have to be provided with every block.
- The relative amount of non-data space necessary for a block scheme is higher than for a sector-scheme.

The Cost of a Disk Access

- The time to access a sector in a track on a surface is divided into 3 components:

Time Component	***Action***
Seek Time	Time to move the read/write arm to the correct cylinder
Rotational delay (or latency)	Time it takes for the disk to rotate so that the desired sector is under the read/write head
Transfer time	Once the read/write head is positioned over the data, this is the time it takes for transferring data

Seek time

- **Seek time** is the time required to move the arm to the correct cylinder.
- Largest in cost.

Typically:

  – 5 ms (miliseconds) to move from one track to the next (track-to-track)
  – 50 ms maximum (from inside track to outside track)
  – 30 ms average (from one random track to another random track)

Latency (rotational delay)

- **Latency** is the time needed for the disk to rotate so the sector we want is under the read/write head.
- Hard disks usually rotate at about 5000 rpm, which is one revolution per 5 msec.
- **Note**:
  - Min latency = 0
  - Max latency = Time for one disk revolution
  - **Average latency (*r*)** = (min + max) / 2
    = max / 2
    = time for ½ disk revolution
- Typically, 2 – 3 ms average

Transfer Time

- **Transfer time** is the time for the read/write head to pass over a block.
- The transfer time is given by the formula:

$$\text{Transfer time} = \frac{\text{number of bytes transferred}}{\text{number of bytes on a track}} \times \text{rotation time}$$

- e.g., if there are 63 sectors per track, the time to transfer one sector would be 1/63 of a revolution.

Sequential and Random Reading

Reading *b* blocks:

i. Sequentially:

$$\underbrace{s + r}_{\text{insignificant for large files}} + b * \text{btt}$$

$$\Rightarrow \quad b * \text{btt}$$

ii. Randomly:

$$b * (s + r + \text{btt})$$

Block **Transfer** Time (**btt**):Time needed to read block to memory (buffer).

Example: t (**data transfer** speed / **data** rate)=3000 bytes/msec

b: denote the number of blocks to be read

Sequential Reading

- Given the following disk:
  - $s = 16$ ms
  - $r = 8.3$ ms
  - Block transfer time = 8.4 ms

a) Calculate the time to read 10 sequential blocks

b) Calculate the time to read 100 sequential blocks

Random Reading

Given the same disk,

a) Calculate the time to read 10 blocks randomly

b) Calculate the time to read 100 blocks randomly