Analysis of Algorithms 1
(Fall 2013)
Istanbul Technical University
Computer Eng. Dept.

Secondary Storage Devices
READING ASSIGNMENT

Course slides from Binnur Kurt's File Organization course.

Outline

- Secondary storage devices
- Magnetic Disc
- (Tape, CD, DVD)
- Buffer Management

Secondary Storage Devices

- Secondary storage (external memory) is slower and larger than main memory. We will go through different secondary storage devices in order to understand how they work .
- Two major types of storage devices
  - Direct Access Storage Devices (DASDs)
    - Magnetic Disks

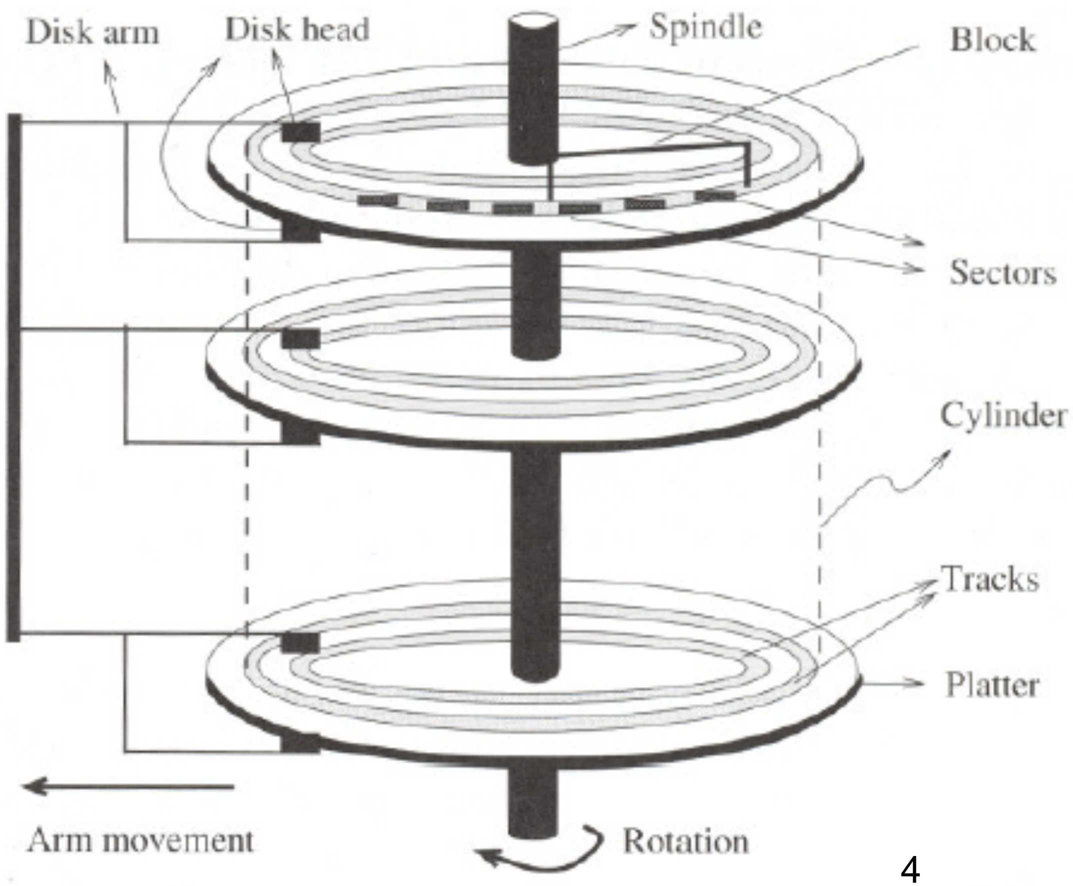
      Hard Disks (high capacity, low cost per bit)
    - Optical Disks

      CD-ROM,DVD-ROM

      (Read-only/write-once, holds a lot of data, cheap, CD-ROMs that you can add on data are now available)
  - Serial Devices
    - Magnetic Tapes (primarily used for backups)

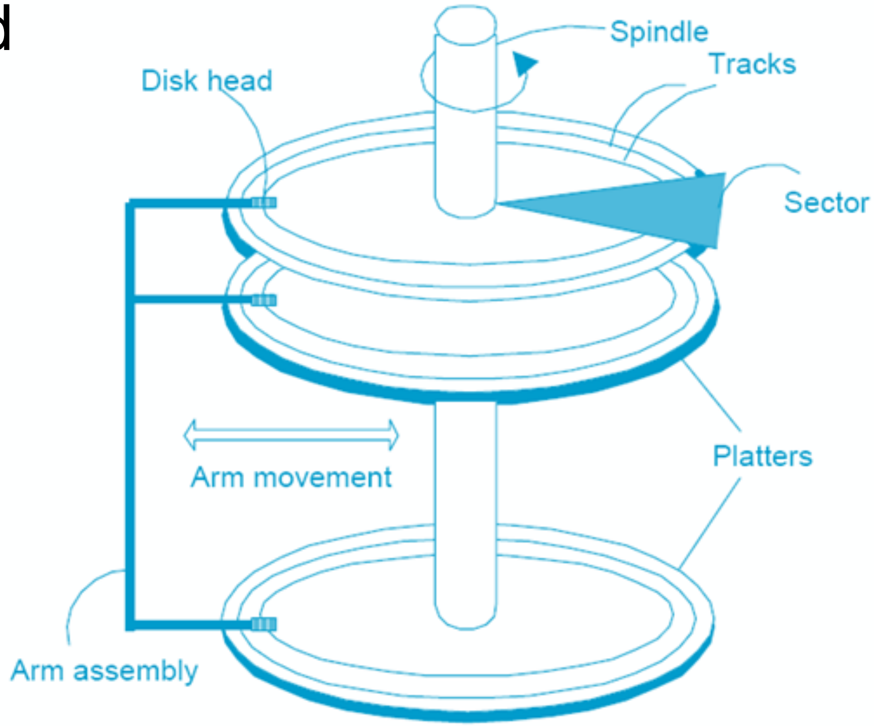
Magnetic Disks

- Magnetic disks support direct access to a desired location
- Simplified structure of a disk
  - Disk blocks
  - Tracks
  - Platters
  - Cylinder
  - Sectors
  - Disk heads
  - Disk Controller
  - Seek Time
  - Rotational delay



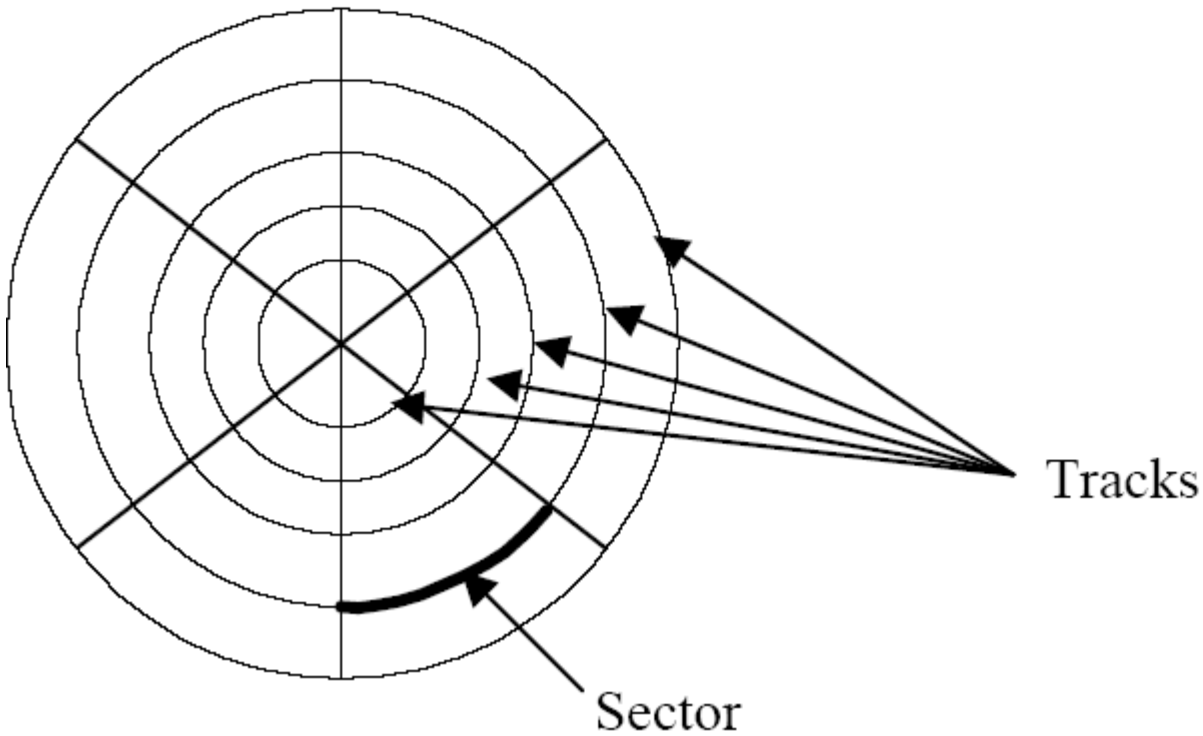
Components of a Disk

- The platters spin (7200 rpm)
- The arm assembly is moved in or out to position a head on a desired track. Tracks under heads make a *cylinder* (imaginary!).
- Only one head reads/writes at any one time
- *Block size* is a multiple of *sector size* (which is fixed)



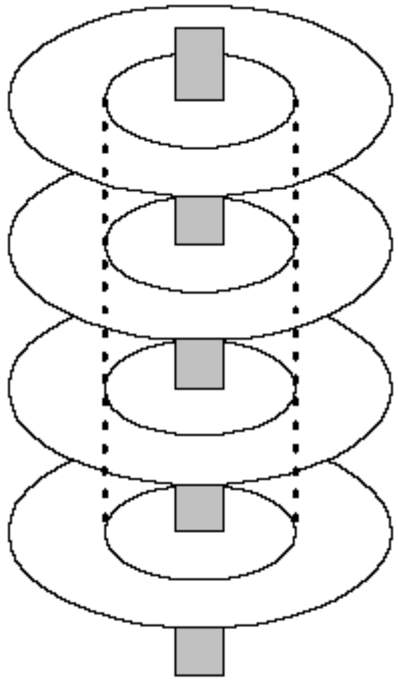
Looking at a Surface

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



Cylinder

- Cylinder: the set of tracks on a disk that are directly above/ below each other
- All the information on a cylinder can be accessed without moving the read/write arm (seeking)

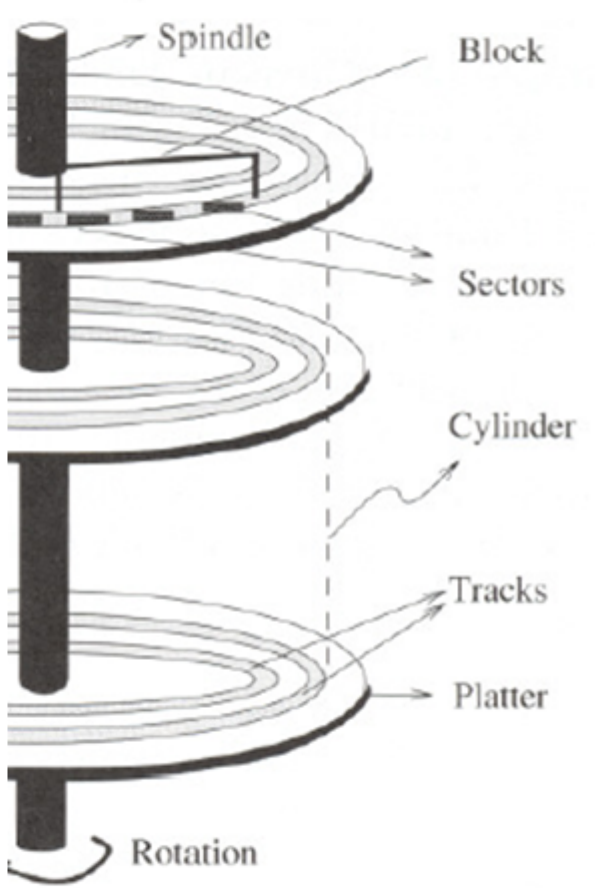


The Bottleneck

- 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.

How to Calculate Disk Capacity

- Number of cylinders = number of tracks in a surface
- Track capacity = number of sector per track × bytes per sector
- Cylinder capacity = number of surfaces × track capacity
- Drive capacity = number of cylinders × cylinder capacity



Disk as a Bottleneck

- Processes are often disk-bound
- network and CPU have to wait a long time for the disk to transmit data
- Various techniques to solve this problem

1. **Multiprocessing**: (CPU works on other jobs while waiting for the disk)
2. **Disk Striping**:
   - ► Putting different blocks of the file in different drives.
   - ► Independent processes accessing the same file may not interfere with each other (parallelism)
3. **RAID** (Redundant Array of Independent Disks)
4. **RAM** Disk (Memory Disk) Piece of main memory is used to simulate a disk (speed *vs.* volatility)

Disk as a Bottleneck (cont.)

- Various techniques to solve this problem

5. Disk Cache:

- ► Large block of memory configured to contain pages of data from a disk.
- ► When data is requested from disk, first the cache is checked.
- ► If data is not there (miss) the disk is accessed

BUFFER MANAGEMENT

A journey of a byte

- Suppose in our program we wrote:

```
outfile << c;
```

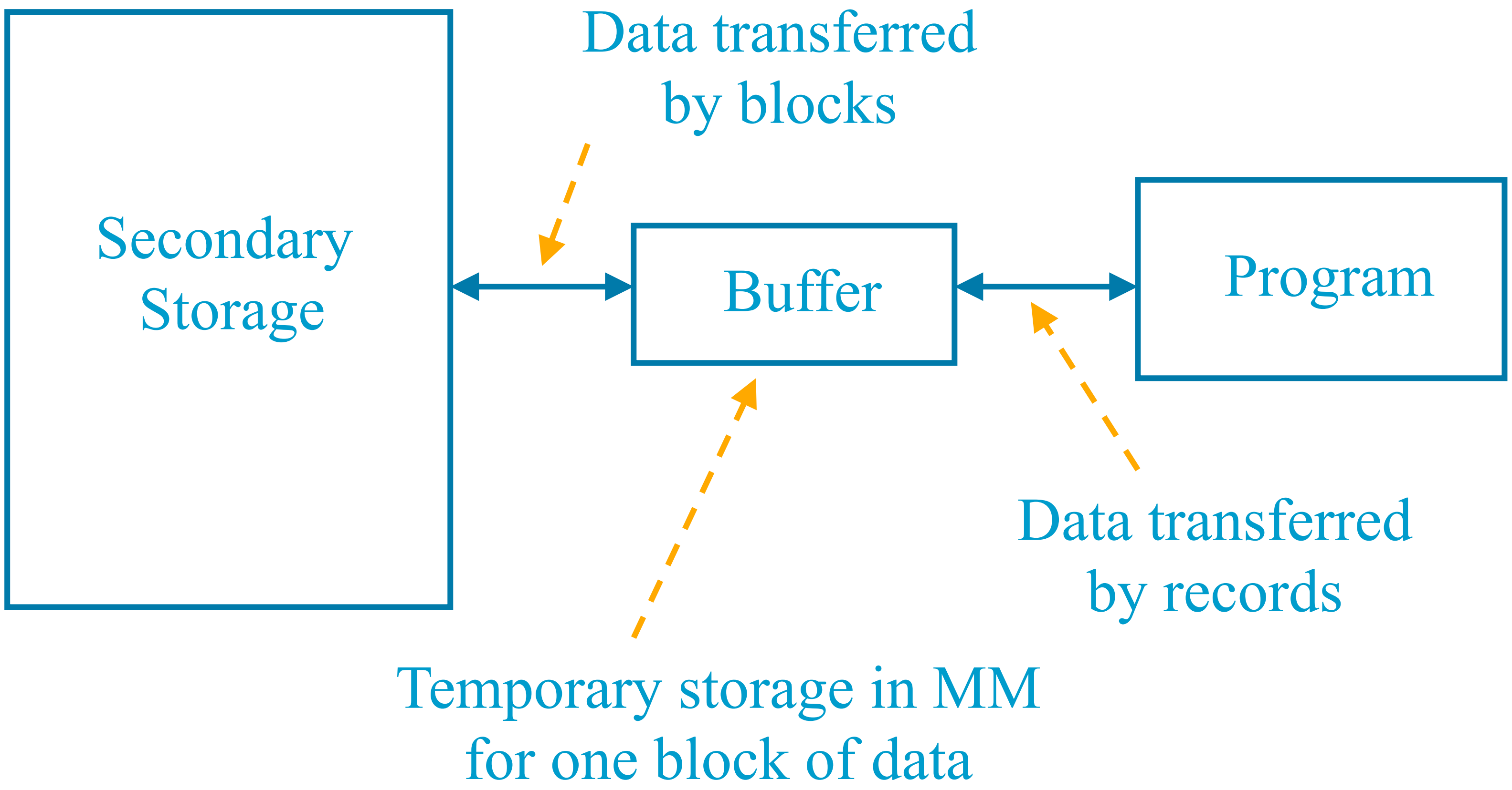
- This causes a call to the **file manager** (a part of O.S. responsible for I/O operations)
- The O/S (file manager) makes sure that the byte is written to the disk.
- Pieces of software/hardware involved in I/O:
  - Application Program
  - Operating System/ File Manager
  - I/O Processor
  - Disk Controller

- **Application Program**
  - Requests the I/O operation
- **Operating System / File Manager**
  - Keeps tables for all opened files
  - Brings appropriate sector to buffer.
  - Writes byte to buffer
  - Gives instruction to I/O processor to write data from this buffer into correct place in disk.
  - Note: the buffer is an exact image of a cluster in disk.
- **I/O Processor**
  - Separate chip; runs independently of CPU
  - Find a time when drive is available to receive data and put data in proper format for the disk
  - Sends data to disk controller
- **Disk controller**
  - Separate chip; instructs the drive to move R/W head
  - Sends the byte to the surface when the proper sector comes under R/W head.

Buffer Management

- Buffering means working with large chunks of data in main memory so the number of accesses to secondary storage is reduced.
- We'll discuss the System I/O buffers. These are beyond the control of application programs and are manipulated by the O.S.
- Note that the application program may implement its own "buffer" – i.e. a place in memory (variable, object) that accumulates large chunks of data to be later written to disk as a chunk.

System I/O Buffer



Buffer Bottlenecks

- Consider the following program segment:

```
while (1) {
  infile >> ch;
   if (infile.fail()) break;
   outfile << ch;
}
```

- What happens if the O.S. used only one I/O buffer?

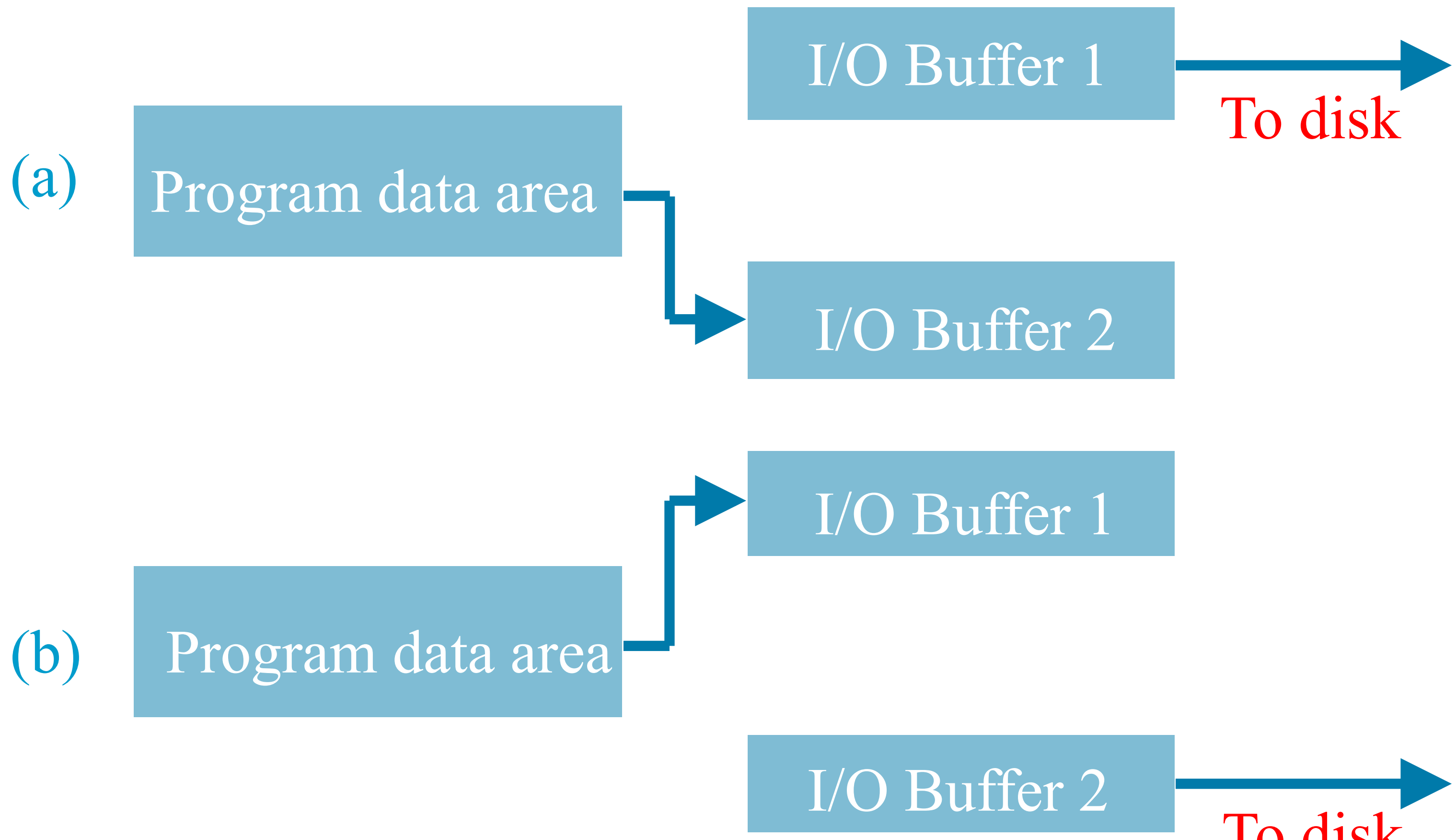
  ⇒ Buffer bottleneck

- Most O.S. have an input buffer and an output buffer.

Buffering Strategies

- **Double Buffering:** Two buffers can be used to allow processing and I/O to overlap.
  - Suppose that a program is only writing to a disk.
  - CPU wants to fill a buffer at the same time that I/O is being performed.
  - If two buffers are used and I/O-CPU overlapping is permitted, CPU can be filling one buffer while the other buffer is being transmitted to disk.
  - When both tasks are finished, the roles of the buffers can be exchanged.
- The actual management is done by the O.S.

Double Buffering



Other Buffering Strategies

- <u>Multiple Buffering</u>: instead of two buffers any number of buffers can be used to allow processing and I/O to overlap.
- <u>Buffer pooling</u>:
  - There is a pool of buffers.
  - When a request for a sector is received, O.S. first looks to see that sector is in some buffer.
  - If not there, it brings the sector to some free buffer. If no free buffer exists, it must choose an occupied buffer. Usually LRU (Least Recently Used) strategy is used.