4 Secondary Storage Devices

Content

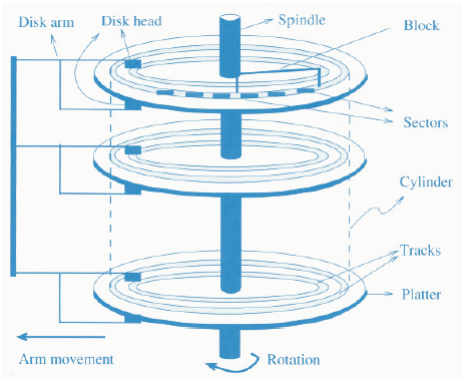
- Secondary storage devices
- Organization of disks
- Organizing tracks by sector
- Organizing tracks by blocks
- Non-data overhead
- The cost of a disk access
- Disk as a bottleneck

Secondary Storage Devices

- Since secondary storage is different from main memory we have to understand how it works in order to do good file designs.
- 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)
  - Serial Devices
    - Magnetic Tapes

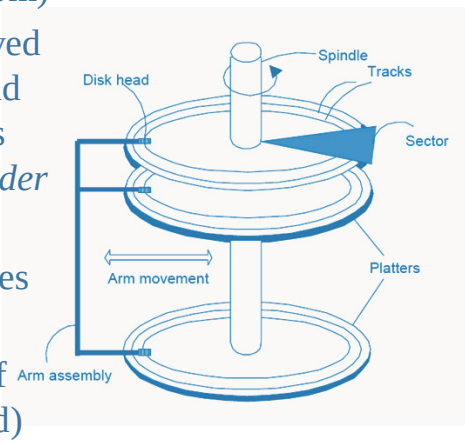
Magnetic Disks

- 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



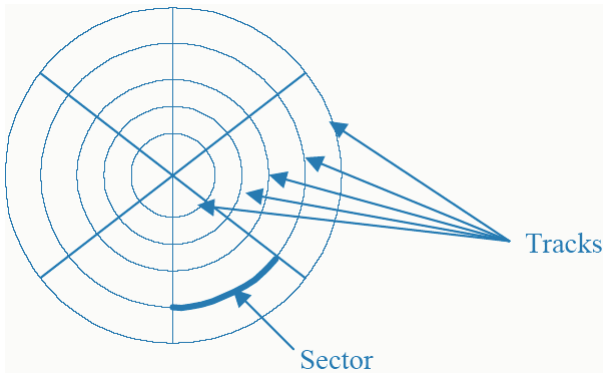
Components of a Disk

- 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)



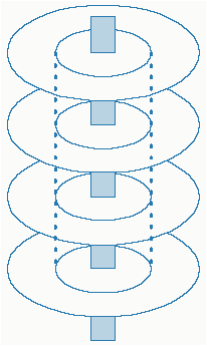
Looking at a Surface

- 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

An Example

- We have fixed-length records
- Number of records = 50.000 records
- Size of a record = 256 bytes
- Disk characteristics
  - Number of bytes per sector = 512
  - Number of sectors per track = 63
  - Number of tracks per cylinder = 16
  - Number of cylinders = 4092
- How many cylinders are needed?

Clusters, Extents and Fragmentation

- The file manager is the part of the operating system responsible for managing files
- The file manager maps the logical parts of the file into their physical location
- A **cluster** is a fixed number of contiguous sectors
- The file manager allocates an integer number of clusters to a file. An example: Sector size: 512 bytes, Cluster size: 2 sectors
  - If a file contains 10 bytes, a cluster is allocated (1024 bytes).
  - There may be unused space in the last cluster of a file. This unused space contributes to **internal fragmentation**

More on Clusters

- Clusters are good since they improve sequential access: reading bytes sequentially from a cluster can be done in one revolution, seeking only once.
- The file manager maintains a file allocation table (FAT) containing for each cluster in the file and its location in disk
- An extent is a group of contiguous clusters. If file is stored in a single extent then seeking is done only once.
- If there is not enough contiguous clusters to hold a file, the file is divided into 2 or more extents.

Fragmentation

- Due to records not fitting exactly in a sector
  - Example: Record size = 200 bytes, sector size = 512 bytes
  - to avoid that a record span 2 sectors we can only store 2 records in this sector (112 bytes go unused per sector
  - the alternative is to let a record span two sectors, but in this case two sectors must be read when we need to access this record)
- Due to the use of clusters
  - If the file size is not multiple of the cluster size, then the last cluster will be partially used.

How to Choose Cluster Size

- Some OS allow the system administrator to choose the cluster size.
- When to use **large cluster size**?
  - When disks contain large files likely to be processed sequentially.
  - Example: Updates in a master file of bank accounts (in batch mode)
- What about **small cluster size**?
  - When disks contain small files and/or files likely to be accessed randomly
  - Example : online updates for airline reservation

Organizing Tracks By Blocks

- Disk tracks may be divided into user-defined blocks rather than into sectors.
- The amount transferred in a single I/O operation can vary depending on the needs of the software designer
- A block is usually organized to contain an integral number of logical records.
- Blocking Factor = number of records stored in each block in a file
- No internal fragmentation, no record spanning two blocks

SubBlocks

- A block typically contains subblocks:
- Count subblock: contains the number of bytes in a block
- Key subblock (optional): contains the key for the last record in the data subblock (disk controller can search can search for key without loading it in main memory)
- Data subblock: contains the records in this block.

NonData Overhead

- Amount of space used for extra stuff other than data
- **Sector-Addressable Disks**
  - at the beginning of each sector some info is stored, such as sector address, track address, condition (if sector is defective);
  - there is some gap between sectors
- **Block-Organized Disks**
  - subblocks and interblock gaps is part of the extra stuff; more nondata overhead than with sector-addressing.

An Example

- Disk characteristics
  - Block-addressable Disk Drive
  - Size of track = 20.000 bytes
  - Nondata overhead per block = 300 bytes
- File Characteristics
  - Record size = 100 bytes
- How many records can be stored per track for the following blocking factors?
  1. Block factor = 10
  2. Block factor = 60

Solution for the Example

- Blocking factor is 10
- Size of data subblocks = 1000
- Number of blocks that can fit in a track =

$$\left\lfloor \frac{20000}{1300} \right\rfloor = \lfloor 15.38 \rfloor = 15$$

- Number of records per track = 150 records

Solution for the Example

- Blocking factor is 60
- Size of data subblocks = 6000
- Number of blocks that can fit in a track =

$$\left\lfloor \frac{20000}{6300} \right\rfloor = \lfloor 3.17 \rfloor = 3$$

- Number of records per track = 180 records

Accessing a Disk Page

- Time to access (read/write) a disk block
  - *seek time* (moving arms to position disk head on track)
  - *rotational delay* (waiting for block to rotate under head)
  - *transfer time* (actually moving data to/from disk surface)
- Seek time and rotational delay dominate
  - Seek time varies from 1 to 20 msec
  - Rotational delay varies from 1 to 10 msec
  - Transfer rate is about 1msec fro 4KB page
- Key to lower I/O cost: reduce seek/rotation delays: Hardware vs. Software solutions?

An Example of a Current Disk

- Model Seagate ST3200822A
- Capacity 200GB
- Transfer Rate
  - Maximum Internal 683Mbits/sec
  - Maximum External 100Mbytes/sec
- Discs/Heads 2/4
- Bytes Per Sector 512
- Spindle Speed 7200 rpm
- Average Seek 8.5 milliseconds
- Average Latency 4.16 milliseconds

What is the average time to read one Sector?

- Transfer time = revolution time / #sectors per track

$$\frac{\left(\frac{1}{7200}\right)}{170} = \frac{\left(\frac{60}{7200}\right)}{170} = \frac{6}{720 \times 170} = \frac{6}{122400} \cong 0.05 m\,\text{sec}$$

- Average totaltime = average seek time +
  average rottional delay +
  transfer time

  8.5 + 4.16 + 0.05 = 12.71 msec

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 (Con't)

- 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

RAID

- Disk Array: Arrangement of several disks that gives abstraction of a single, large disk.
- Goals: Increase performance and reliability.
- Two main techniques
  - Data striping: Data is partitioned; size of a partition is called the striping unit. Partitions are distributed over several disks.
  - Redundancy: More disks → more failures. Redundant information allows reconstruction of data if a disk fails.

RAID Levels

- Level 0: No redundancy
- Level 1: Mirrored (two identical copies)
  - Each disk has a mirror image (check disk)
  - Parallel reads, a write involves two disks
  - Maximum transfer rate=transfer rate of one disk
- Level 0+1: Striping and Mirroring
  - Parallel reads, a write involves two disks
  - Maximum transfer rate = aggregate bandwidth

RAID Levels

- Level 3: Bit-Interleaved Parity
  - Striping Unit: One bit. One Check Disk.
  - Each read and write request involves all disks; disk array can process one request at a time.
- Level 4: Block-Interleaved Parity
  - Striping Unit: One Disk Block. One Check Disk.
  - Parallel reads possible for small requests, large requests can utilize full bandwidth
  - Writes involve modified block and check disk
- Level 5: Block-Interleaved Distributed Parity
  - Similar to RAID Level 4, but parity blocks are distributed over all disks