File Organization

Lecture 1

File Structure

is a combination of representations for data in files and of operations for accessing the data.

allows applications to read, write and modify data.

It might also support finding the data that matches some search criteria or reading through the data in some particular order.

Data processing

involves :

- Storage of data
- Organization of data
- Access to data

Data Structures vs. File Structures

Both involve:

- Representation of Data
- Operations for accessing data

Difference:

- Data Structures deal with data in main memory
- File Structures deal with data in secondary storage device (File)

Main Memory vs. Secondary Storage

Main Memory

- Fast (since electronic)
- Small (since expensive)

- Volatile (information is lost when power failure occurs)

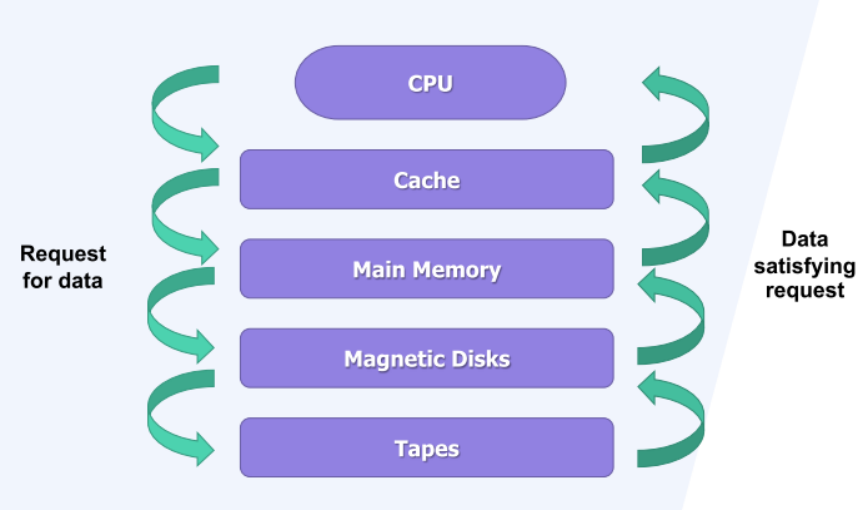
Secondary Storage

- Slow (since electronic and mechanical)
- Large (since cheap)
- Stable, persistent (information is preserved longer)

Main memory : ~120 nanoseconds = 120×10^{-9}

Magnetic Disks : ~30 milliseconds = 30×10^{-6}

Memory Hierarchy



Good File Structure Design

- Fast access to great capacity
- Reduce the number of disk accesses
- By collecting data into buffers, blocks or buckets
- Manage growth by splitting these collections

History of File Structures Design

1. In the beginning... it was the tape
    - Sequential access

- Access cost proportional to size of file [Analogy to sequential access to array data structure]

2. Disks became more common

   Great if index fits into main memory. As file grows we have the same problem we had with a large primary file

   - Direct access [Analogy to access to position in array]
   - Indexes were invented
     - list of keys and points stored in small file
     - allows direct access to a large primary file
3. Tree Structures emerged for main memory(1960's)
   - Binary search trees (BSTs)
   - Balanced, self adjusting BSTs: e.g. AVL trees (1963)
4. A tree structure suitable for files was invented
   - B trees (1979) and B+ trees
   - good for accessing millions of records with 3 or 4 disk accesses
5. What about getting info with a single request?
   - Hashing Tables (Theory developed over 60's and 70's but still a research topic)
     - Good when files do not change too much in time.
   - Expandable, dynamic hashing (late 70's and 80's)
     - One or two disk accesses even if file grows dramatically

Computer File (simply a file)

named collection of data that exists on a storage medium

can contain a group of records, a document ...etc.

Rules for naming files

- Every file has a name and might also have a file extension.
- you must provide a valid file name that adheres to specific rules, referred to as file-naming conventions.

    - Each operating system has a unique set of file-naming conventions.
- Some operating systems also contain a list of reserved words that are used as commands or special identifiers. You cannot use these words alone as a file name.
- You can also use spaces in file names

File (name) Extension

optional file identifier that is separated from the main file name by a period

provide clues to a file's contents

File's Location

- You can store files on a hard drive, removable storage, a network computer, or cloud-based storage.
- To determine a file's location, you must first specify the device where the file is stored.
    - When working with Windows, each local storage device is identified by a device letter. The main hard disk drive is referred to as drive C:
    - Macs do not use drive letters. Every storage device has a name. The main hard disk is called Macintosh HD, for example.
- A disk partition is a section of a hard disk drive that is treated as a separate storage unit.
- Every storage device has a directory containing a list of its files.
    - The main directory is referred to as the root directory. On a PC, the root directory is identified by the device letter followed by a backslash (C:\).
    - A root directory can be subdivided into smaller lists. Each list is called a subdirectory

File Format

organization and layout of data that is stored in a file

The format of a file usually includes :

- header

    - section of data at the beginning of a file that contains information about a file, such as (created, last updated, size, file type)
  - data
  - (possibly) end-of-file marker

Although a file extension is a good indicator of a file's format, it does not really define the format.

Why I can't open some files

- The file might have been damaged by a transmission or disk error.
- Someone might have accidentally changed the file extension.
- Some file formats exist in several variations, and your software might not have the capability to open a particular variation of the format.

File Management

encompasses any procedure that helps you organize your computer-based files so that you can find and use them more efficiently.

File management utilities often use some sort of storage metaphor to help you visualize and mentally organize the files on your disks. (Filling Cabinet, Tree Structure)

File Management Tips

- Use descriptive names
- Maintain file extensions.
- Group similar files.
- Organize your folders from the top down.
- Consider using default folders.
- Use Public folders for files you want to share.
- Do not mix data files and program.
- Don't store files in the root directory.
- Access files from the hard disk.
- Follow copyright rules.

- Delete or archive files you no longer need.
- Back up!

Lecture 2

Secondary Storage Devices

types:

- Direct Access Storage Devices
  - Magnetic Disks
  - optical Disks
- Serial Devices
  - Magnetic Tapes

Data Storage System

components:

- storage medium
- storage device

types:

- magnetic
  - stores data by magnetizing microscopic particles on a disk or tape surface
  - The particles retain their magnetic orientation until that orientation is changed, thereby making disks fairly permanent but modifiable storage media.
- optical
- solid-state

Hard Disk

contains one or more platters and their associated read write heads.

A hard disk platter is a flat, rigid disk made of aluminum or glass and coated with magnetic iron oxide particles.

More platters mean more data storage capacity. The platters rotate as a unit on a spindle, making thousands of rotations per minute (RPM).

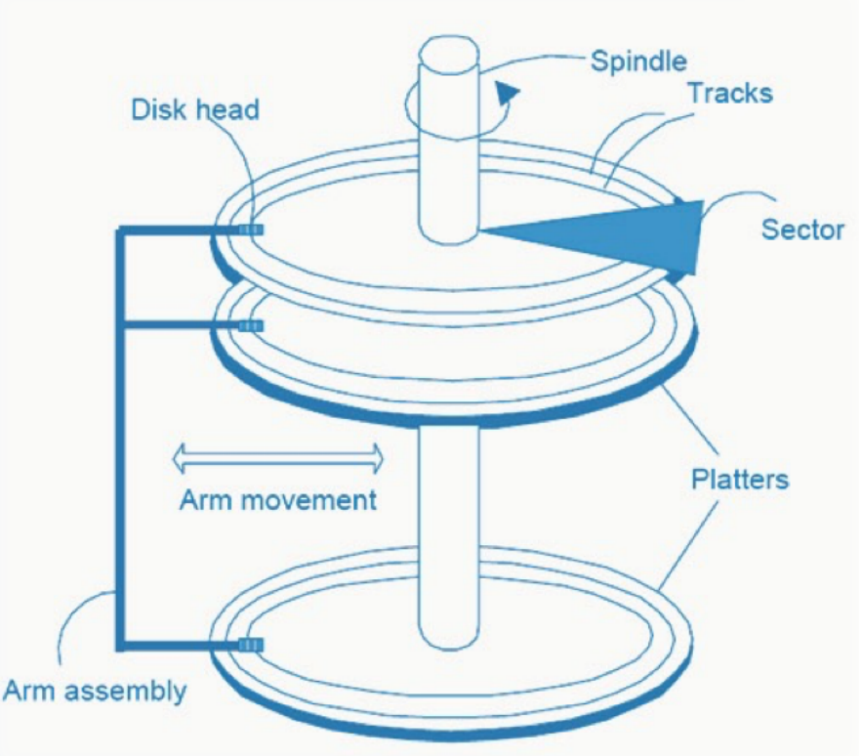
Hard disk platters are typically 3.5" in diameter, with storage capacities ranging from 40 GB to 2 TB.

Each platter has a read-write head that hovers just a few microinches above the surface.

A read-write head mechanism in the disk drive magnetizes particles to write data, and senses the particles' polarities to read data.

Hard disk technology is the preferred type of main storage for most personal computers because:

- It provides lots of storage capacity.
- It provides fast access to files.
- economical.



simplified structure of a disk:

- Rotational Delay

- Disk Controller
- Disk Heads
- Platters
- Seek Time
- Cylinder
  - the set of tracks on a disk that are directly above/below each other
  - Tracks under heads make a cylinder.
- Tracks
- Disk blocks
- Sectors

The arm assembly is moved in or out to position a head on a desired track.

Only one head reads/writes at any one time

All the information on a cylinder can be accessed without moving the read/write arm (seeking)

The bottleneck of a disk access is moving the read/write arm.

- 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.

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

File Manager

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

The file manager allocates an integer number of clusters to a file.

There may be unused space in the last cluster of a file. so This unused space contributes to internal fragmentation

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

Extent

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

Blocks

Disk tracks may be divided into user-defined blocks rather than into sectors

A block is usually organized to contain an integral number of logical records.

No internal fragmentation, no record spanning two blocks

Blocking Factor = number of records stored in each block in a file

contain 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 for key without loading it in main memory)
- Data subblock: contains the records in this block.

Non-data 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

Cost of Disk Access

Seek Time

the time required to move the access arm to position disk head on the correct track.

Rotational Delay

the time it takes for the disk to rotate so the desired sector is under the read/write head

Transfer Time

the amount of time required to move data to/from disk surface.

Average Total Time

Average Seek time + Average Rotational delay + Transfer time

Solid State Storage

stores data in erasable, rewritable circuitry, rather than on spinning disks or streaming tape

- Fast access to data because it includes no moving parts.
- Very durable; it is virtually impervious to vibration, magnetic fields, or extreme temperature fluctuations

Lecture 3

Journey of a Byte

1. The program asks the operating system to write the contents of the variable `ch` to the next available position in the file.
2. The operating system passes the job on to the file manager

3. The file manager looks up the file in a table containing information about it, such as whether the file is open and available for use, what types of access are allowed, if any, and what physical file the logical name corresponds to.
4. The file manager searches a file allocation table for the physical location of the sector that is to contain the byte.
5. The file manager makes sure that the last sector in the file has been stored in a system I/O buffer in RAM, then deposits the ‘p’ into its proper position in the buffer.
6. The file manager gives instructions to the I/O processor about where the byte is stored in RAM and where it needs to be sent on the disk.
7. The I/O processor, finds a time when the drive is available to receive the data and puts the data in proper format for the disk. It may also buffer the data to send it out in chunks of the proper size for the disk.
8. The I/O processor sends the data to the disk controller.
9. The controller instructs the drive to move the read/write head to the proper track, waits for the desired sector to come under the read/write head, then sends the byte to the drive to be-deposited, bit-by-bit, on the surface of the disk.

Application program

- Requests the I/O operation

OS/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.

I/O Processor

- A separate chip; runs independently of CPU
- Finds 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

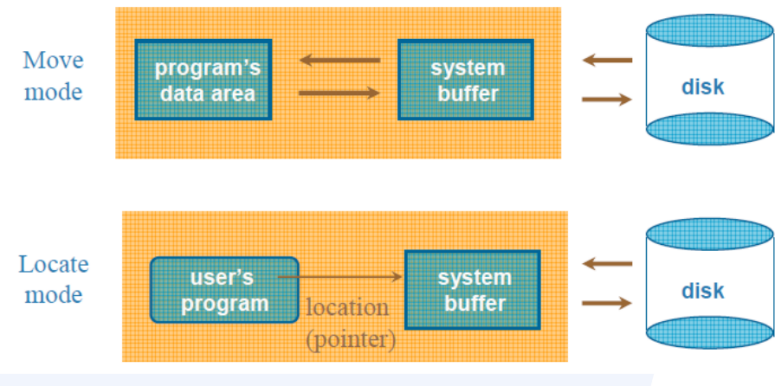
- A separate chip; instructs the drive to move R/W head
- Sends the byte to the surface when the proper sector comes under R/W head.

Buffer Management

Buffering means working with large chunks of data in main memory so the number of accesses to secondary storage is reduced.

Buffer Strategies

- Double Buffering: Two buffers can be used to allow processing and I/O to overlap
- Multiple Buffering: instead of two buffers any number of buffers can be used to allow processing and I/O to overlap
- Buffer pooling:
- Move Mode (using both system buffer & program buffer)
    - moving data from one place in RAM to another before they can be accessed
    - sometimes, unnecessary data moves
- Locate Mode (using system buffer only or program buffer only)
    - perform I/O directly between secondary storage and program buffer (program's data area)
    - system buffers handle all I/Os, but program uses locations through pointer variable



Field

a data value, smallest unit of data with logical meaning.

a file can be treated as:

1. a stream of bytes
2. a collection of records with fields

methods for organizing fields are

1. Fix the length of fields
    - Adv: Easy to Read/Store
    - DisAdv: Waste space with padding
2. Begin Each Field with a Length Indicator
    - Adv: Easy to jump ahead to the end of the field
    - DisAdv: Long fields require more than 1 byte to store length (Max is 255)
3. Separate the Fields with Delimiters
    - Adv: May waste less space than with length-based
    - DisAdv: Have to check every byte of field against the delimiter
4. Use a “Keyword=Value” Expression to Identify Fields
    - Adv: Fields are self describing allows for missing fields
    - DisAdv: Waste space with keywords

Record

A group of fields that forms a logical unit.

3. Begin each record with a length indicator
4. Use an index to keep track of addresses
5. Place a delimiter at the end of each file

methods for organizing records are

1. Fix the length of fields (Make records a predictable number of bytes)
2. Make records a predictable number of fields
3. Begin each record with a length indicator
4. Use an index to keep track of addresses
5. Place a delimiter at the end of each record

Lecture 4

Record Access

Record Keys

Key: a subset of the fields in a record used to uniquely identify the record

Primary Key: must identify record uniquely, not dataless, canonical form

Secondary Key: doesn't identify record uniquely, not dataless, canonical form

Keys correspond to fields, or combination of fields, that may be used in a search

Sequential Search

$$worst = O(n)$$
$$Avg = O(n/2)$$

To improve the performance of sequential search use **Record Blocking**

Record Blocking: by reading in a block of several records all at once and then processing that block of records in memory.

Appropriate for

- ASCII files: in which you are searching for some pattern
- Files with few records

- Files that hardly ever need to be searched
- Files in which you want all records with a certain secondary key value, where a large number of matches is expected

Direct Access

Being able to seek directly to the beginning of the record.

$$O(1)$$

possible when we know the Relative Record Number (RRN)

requires records of fixed length

Header Record

record placed at the beginning of a file that is used to store information about the file contents and the file organization

The header record usually has a different structure & different size than the data records in the file.