Operating Systems

Course Code: **71203002004**

Data Access Techniques & Data Integrity Protection

by -

Minal Rajwar

File Access Methods

File access methods are ways an operating system reads or writes data in files. They determine how data is organized, retrieved, and modified. Choosing the right method affects performance and efficiency.

There are three main types of file access:

1. Sequential Access

2. Direct Access

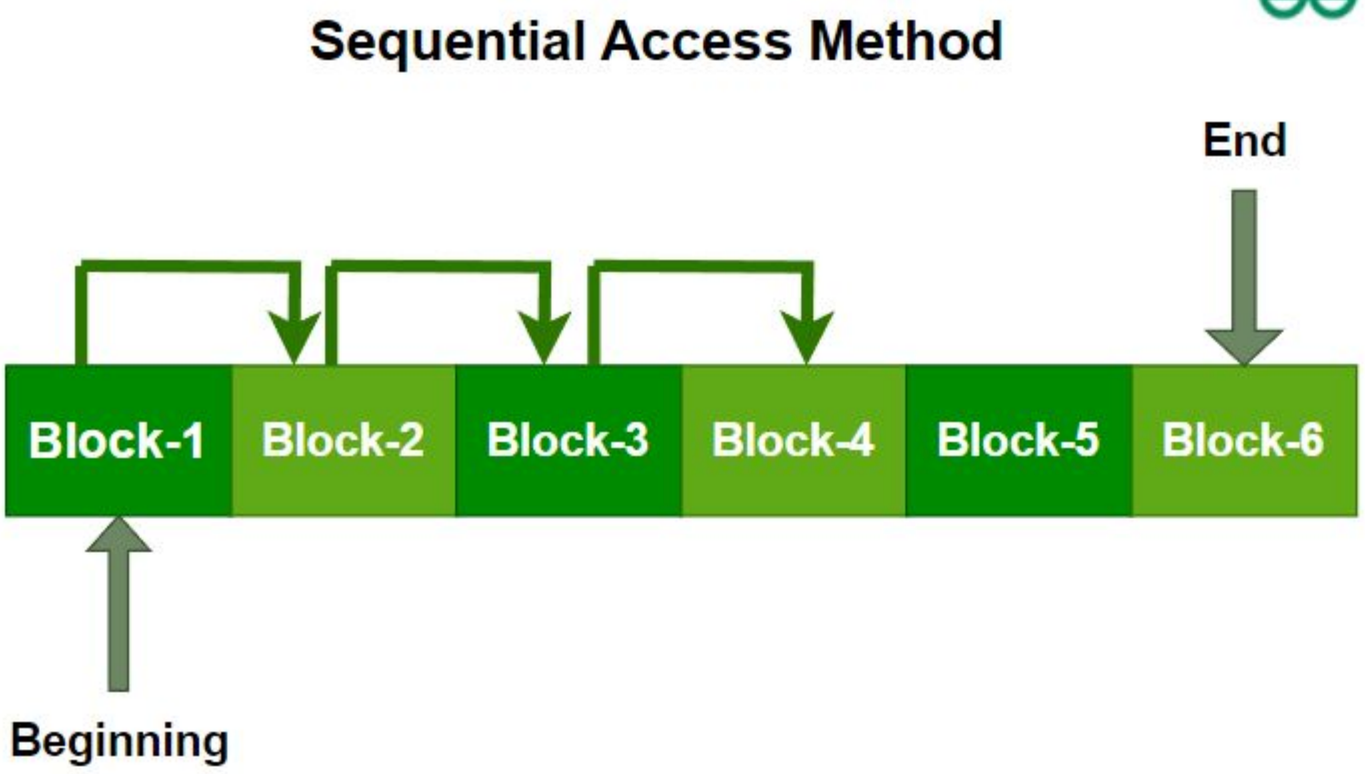
3. Index Sequential Access

1. Sequential Access

- Data is read or written **one record after another**, starting from the beginning.
- The **file pointer moves automatically** after each read or write.

Key Points:

- Accesses records in order.
- `Read` moves the pointer forward by one record.
- `Write` adds data at the end and moves the pointer.
- Commonly used for tapes.


Sequential Access Method
End
Block-1
Block-2
Block-3
Block-4
Block-5
Block-6
Beginning

1. Sequential Access

Advantages:

- Simple to use.
- Less prone to data corruption.
- Efficient for processing records in order.

Disadvantages:

- Slow for searching specific records.
- Hard to insert or update in the middle.
- May waste space if record sizes vary.

2. Direct Access (Random Access)

- Allows **reading or writing any record directly** using its address.
- No need to go through other records.

Advantages:

- Faster access to any record.
- No need to traverse all previous records.

Disadvantages:

- Complex to implement.
- Needs extra storage for addresses or pointers.

3. Index Sequential Access

- Combines **sequential and direct access**.
- Uses an **index** (like a book index) to locate records quickly.

Advantages:

- Fast searching.
- Supports sequential and random access.
- Reduces access time for large files.

Disadvantages:

- Harder to implement.
- Requires extra storage for the index.
- Updates are slower (data + index need updates).

Other File Access Methods

1. Relative Record Access

- Accesses records **relative to the current file pointer**.
- Good for **ordered processing**.

Advantages:

- Allows random access by moving the pointer.
- Faster than sequential for individual records.
- Supports sequential processing too.

Disadvantages:

- Needs fixed-length records.
- Inserting or deleting is difficult.
- Not suitable for frequent updates.

Other File Access Methods

2. Content-Addressable Access (CAA)

- Accesses records **based on content**, not address.
- Uses a **hash function** to generate a unique key for each record.

Advantages:

- Efficient for large databases.
- Flexible insertion and deletion.
- Ensures data integrity.

Disadvantages:

- Extra overhead for hash calculation.
- Possibility of collisions (two records get the same key).
- Limited key space depending on hash size.

Buffering

- **Definition:** Temporary memory space that stores input data when the speed of data arrival and processing differ.
- **Purpose:** Ensures smooth data flow without making the input device wait.
- **Example:** Video streaming (video “buffers” to prevent interruptions) or printer spoolers.

Advantages:

- Smooth and uniform disk access.
- Improves input/output performance.
- Reduces system calls, boosting overall system speed.

Disadvantages:

- Large buffers can slow the system.
- Synchronization issues may occur.

Caching

- **Definition:** Temporary storage of frequently used data in a **fast memory** (like RAM or SSD) to reduce access time.
- **Purpose:** Speeds up data retrieval for repeated operations.
- **Example:** Visiting a website—the second time it loads faster because of cached data.

Advantages:

- Faster data access than from hard disk.
- Improves application performance.
- Reduces server load.

Disadvantages:

- Data is lost when the system is turned off.
- Expensive compared to normal memory.

Journaling

Definition: Keeps file system data and metadata consistent after crashes or power failures.

How it works: Changes are first written to a **journal/log**, then applied to the file system. If a crash occurs, unfinished transactions are replayed from the log.

Types:

1. **Metadata journaling:** Logs only metadata; faster but less safe for data.
2. **Data journaling:** Logs both data and metadata; safer but slower.
3. **Ordered journaling:** Writes data first, then metadata; balances speed and safety.

RAID and Checksums

RAID: Combines multiple disks into one logical unit for **redundancy and fault tolerance**.

- Example: RAID 5 uses parity to recover data if a drive fails.

Checksums: Small values calculated from data to **detect corruption**.

- Used in "checksumming RAID" (e.g., ZFS, Btrfs) to detect and fix silent data corruption.

RAID and Checksums

Feature	RAID	Checksums
Purpose	Protects against drive failure	Protects against data corruption
Detection	May miss silent corruption	Detects and helps correct corruption
Implementation	Hardware/software layer	Integrated with file system

Backup and Recovery

Backup: Making copies of data to protect against loss.

- **Types:**
  - Full: All data.
  - Incremental: Only changed data since last backup.
  - Differential: Changed data since last full backup.
  - System image: Complete OS, apps, and files.
- **Best practice:** Follow **3-2-1 rule**: 3 copies, 2 different media, 1 off-site.

Recovery: Restoring data from backups.

- **Metrics:**
  - **RPO:** Max data loss tolerated.
  - **RTO:** Max downtime tolerated.
- **Types:** File-level to full system recovery; OS tools like system restore or recovery drives help revert systems.

DISCUSSION & REVISION

1. Which file access method reads data one record after another?
2. What stores frequently used data to speed up access?
3. Which journaling type logs both data and metadata?
4. RAID provides protection against what?
5. What is created to restore data after loss or corruption?

REFERENCES

1. https://www.geeksforgeeks.org/operating-systems/file-access-methods-in-operating-system/
2. https://www.geeksforgeeks.org/operating-systems/difference-between-buffering-and-caching-in-os/