UNIT V

Storage and File Systems

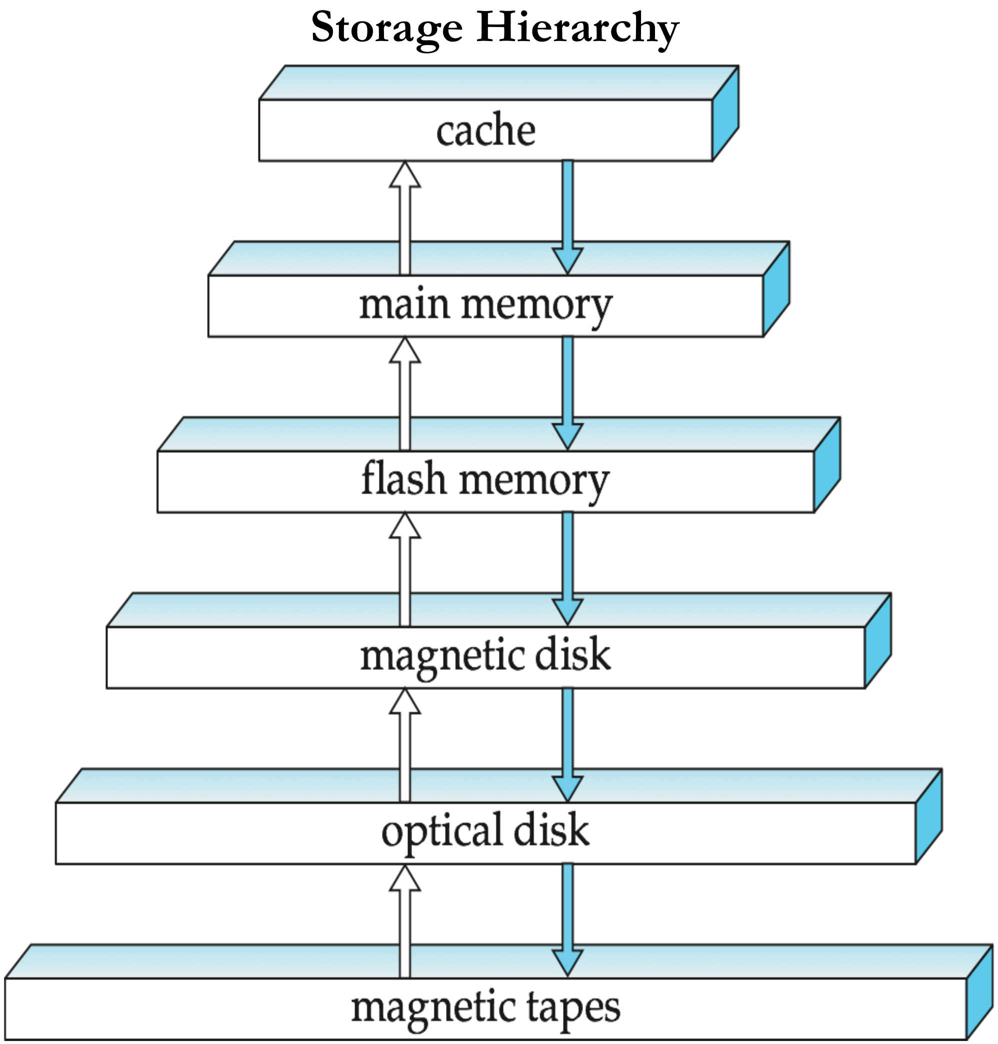
OBJECTIVE:

- Overview of Physical Storage Media
- Magnetic Disks and disk storage
- File Organization
- Static and Dynamic Hashing

Overview of physical storage media

- **Several types of data storage exist in most computer systems**
- **These storage media are classified by**
  - Speed with which data can be accessed
  - Cost per unit of data to buy the medium
  - Medium's reliability
- **volatile storage**: loses contents when power is switched off
- **non-volatile storage**:
  - Contents persist even when power is switched off.
  - Includes secondary and tertiary storage, as well as batter-backed up main-memory.

Overview of physical storage media



Physical Storage Media

- **Cache** – fastest and most costly form of storage; volatile; managed by the computer system hardware.
- **Main memory**: **primary storage** (RAM)
  - fast access
  - generally too small (or too expensive) to store the entire database
  - **Volatile** — contents of main memory are usually lost if a power failure or system crash occurs.
- **Flash memory**
  - Data survives power failure, Data can be written at a location only once, but location can be erased and written to again

Physical Storage Media

- **Magnetic-disk :** Direct Access Storage Devices **(secondary storage)**
  - Primary medium for the long-term storage of data; typically stores entire database.
  - Data must be moved from disk to main memory for access, and written back for storage
  - Much slower access than main memory
  - **direct-access** – possible to read data on disk in any order, unlike magnetic tape
  - Much larger capacity
  - Survives power failures and system crashes
  - disk failure can destroy data, but is rare

Physical Storage Media

- **Optical storage:** Direct Access Storage Devices **(secondary storage)**
  - non-volatile, data is read optically from a spinning disk using a laser
  - CD-ROM (640 MB) and DVD (4.7 to 17 GB) most popular forms
  - Reads and writes are slower than with magnetic disk
- **Tape storage** Serial Devices (**secondary storage)**
  - non-volatile, used primarily for backup (to recover from disk failure), and for archival data
  - **sequential-access** – much slower than disk
  - very high capacity (40 to 300 GB tapes available)

Magnetic Disks components

- Provides the bulk of secondary storage for modern computer systems
- Bits of data (0's and 1's) are stored on circular magnetic platters called disks.
- Each disk platter has a flat, circular shape
- Its two surfaces are covered with a magnetic material and information is recorded in the surfaces
- Platter are made from rigid metal or glass
- Often, several platters are organized into a disk pack (or disk drive).
- Disk surface is divided into concentric **tracks.**
- Tracks are divided into **sectors**
- A **sector** is the smallest addressable unit in a disk.
- Tracks under heads make **a cylinder** (imaginary!).
- A **cylinder** is the set of tracks at a given radius of a disk pack

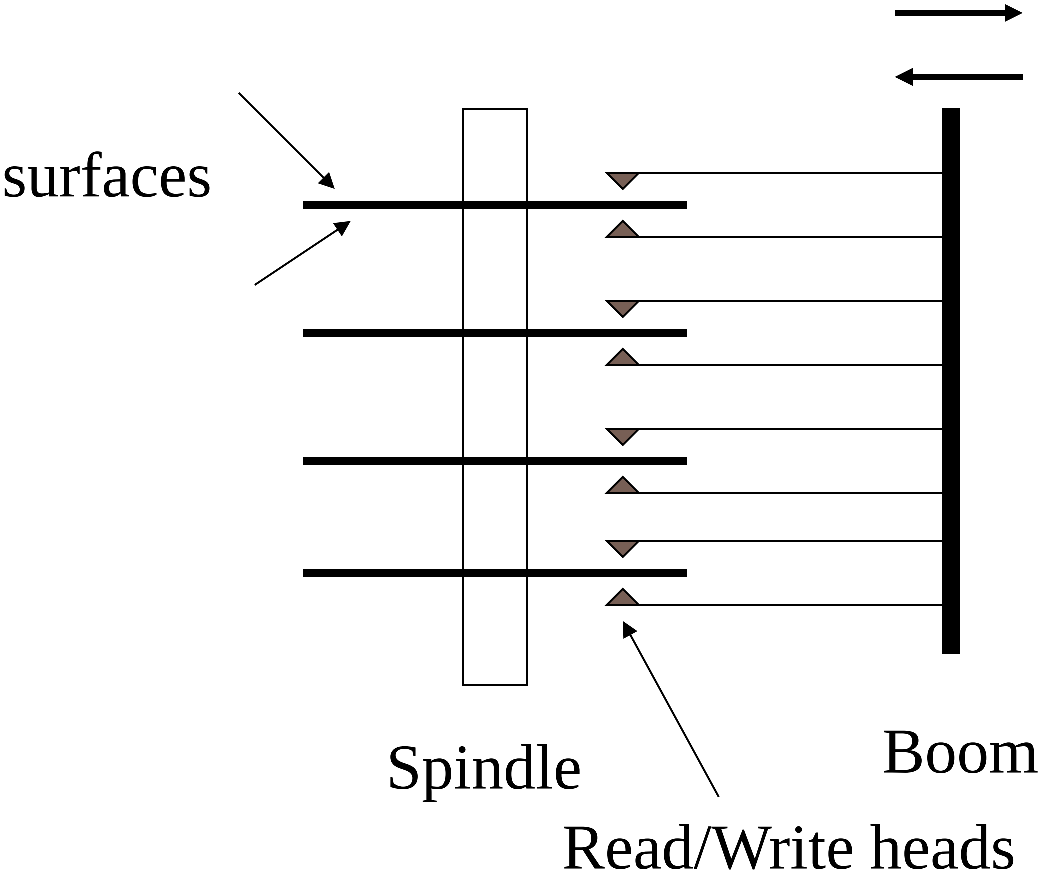
Magnetic Disks operation

- When the disk is in use, a drive motor spins it at a constant high speed (60,90 or 120 revolutions per second)
- A disk head (read-write head) positioned just above the surface of the platter
- A disk head reads and writes bits of data as they pass under the head.
- The read write heads of all the tracks are mounted on a single assembly called a disk arm and move together the disk platters mounted on a spindle and the heads mounted on a disk arm are together known as head disk assemblies
- Heads on all the platters move together

Magnetic Disks operation

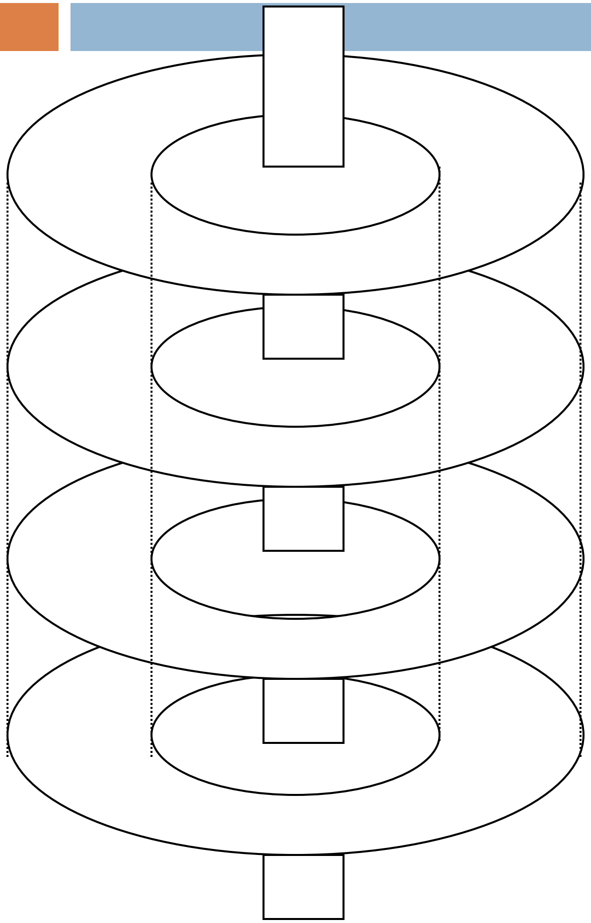
- Disk controllers: typically embedded in the disk drive, which acts as an interface between the CPU and the disk hardware.
- The controller has an internal cache (typically a number of MBs) that it uses to buffer data for read/write requests.
- When a program reads a byte from the disk, the operating system locates the surface, track and sector containing that byte, and reads the entire sector into a special area in main memory called **buffer**.
- It accepts high level commands to read or write a sector and initiates actions such as moving the disk arm to right track and actually reading and writing the data
- 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 on different surfaces, rather than in several tracks on the same surface.

Secondary storage devices :A Disk Drive

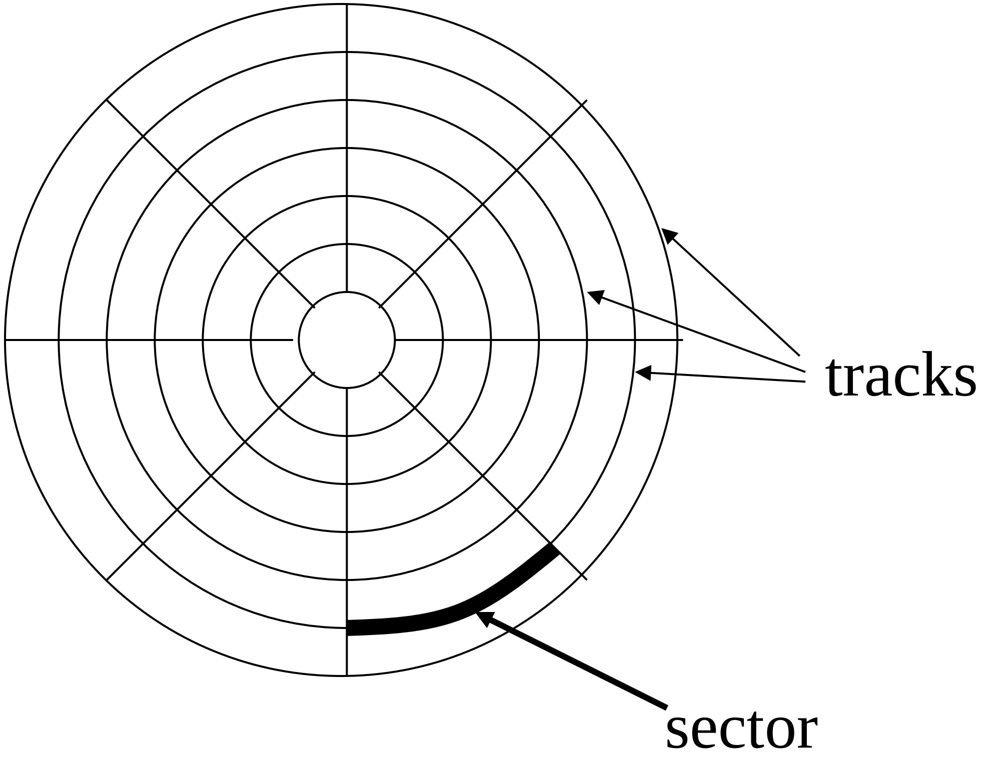


Disk drive with 4 platters and 8 surfaces and 8 RW heads

Secondary storage devices :A Disk Drive

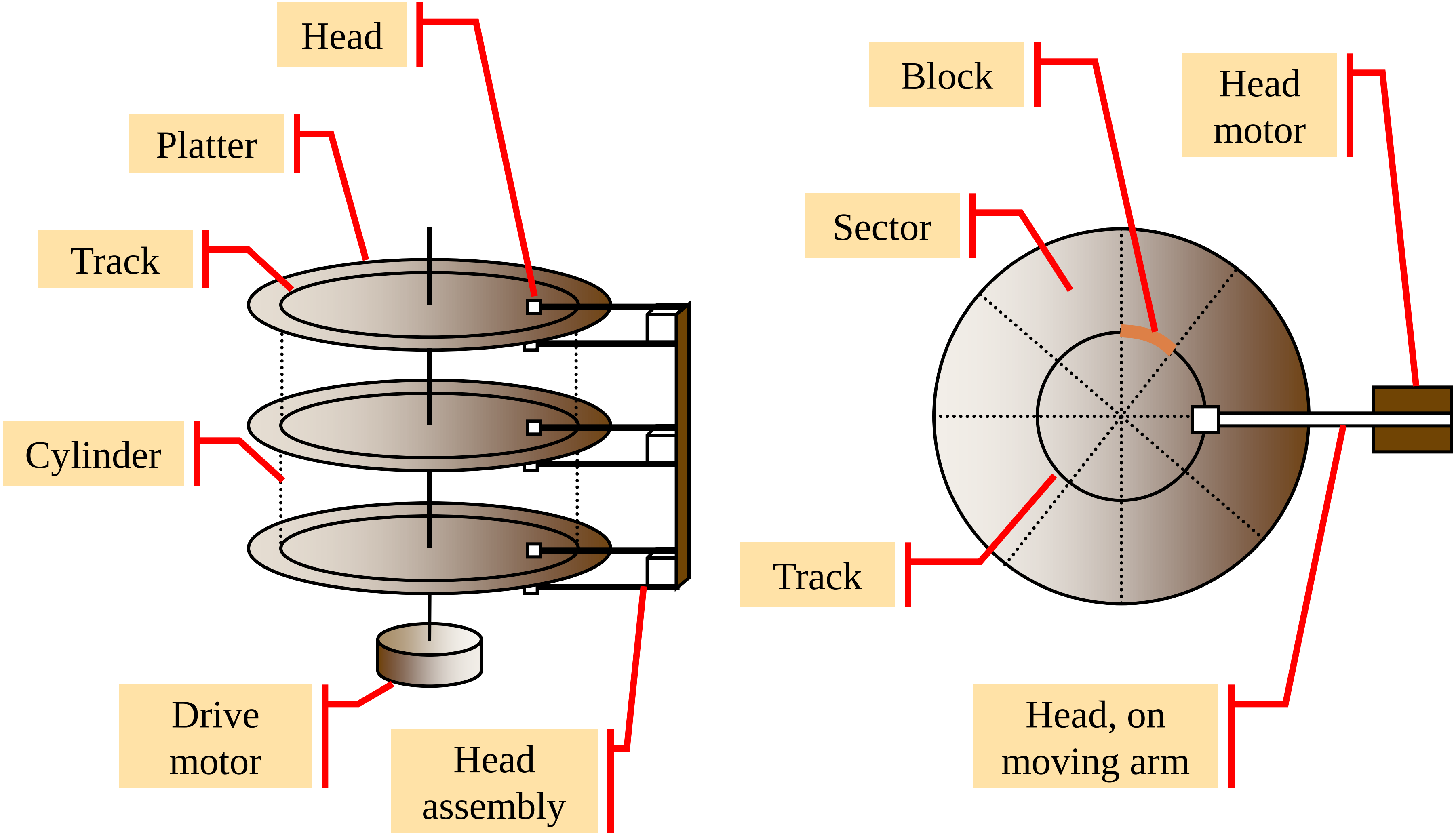


Cylinders



Surface of disk showing tracks and sectors

Hard Disk Layout



Estimating Capacities

- Total Track capacity = no of sectors/track * bytes/sector + no of sectors/track * interblock gap size
- Useful Track capacity= no of sectors/track * bytes/sector
- Cylinder capacity = no of tracks/cylinder * track capacity
- Drive capacity = no of cylinders * cylinder capacity
- Number of cylinders = no 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 = 11

  # 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?

solution

Q. 1 Total memory size= 20000*256 =5120000 bytes

5120000/512 (bytes/sector) =10000 sectors required

10000/40 (sector/track) = 250 tracks required

250/ 11 (no of tracks/cylinder) = 22.72 = 23 cylinders required

Q. 2 Useful Track capacity= no of sectors/track * bytes/sector

= 40 * 512 = 20480 bytes

- □ Cylinder capacity = no of tracks/cylinder * track capacity

  = 11 * 20480 = 225280 bytes

- □ Drive capacity = no of cylinders * cylinder capacity

  = 225280 *1331 = 299847680 bytes = 300 mb

File organization

- File organization refers to the organization of data of a file into records, blocks and access structures
- This includes the way of records and blocks are placed on the storage medium and interlinked
- It is the methodology which is applied to structured computer files.
- An access method provides a group of operations that can be applied to a file.
- Some access methods, can be applied only to files organized in certain ways

File Organization

- Organizing a file depends on what kind of file it happens to be: a file in the simplest form can be a text file,
- Files can also be created as binary or executable types
- Some files may be static- update operations are rarely preformed
- Dynamic files may change frequently

Techniques of File Organization

- The techniques of primary file organization are:
- **Heap files (unordered)**
- **Sorted files (Ordered)**
- **Hashed or Direct files**
- Additional auxiliary access structures called **indexes** which are used to speed up the retrieval of records in response to certain search conditions

Heap files (Unordered Records)

- Records are placed in the files in the order in which they are inserted
- So new records are inserted at the end of the file
- Such organization is called heap file
- It is the simplest and most basic type of organization
- It is also used to collect and store data records for future use
- Though inserting a new record is very efficient, searching for a record (linear search) is an expensive procedure

Heap files (Unordered Records)

- While inserting a new record, the last block of the file is copied into a buffer, the new record is added and the block is then rewritten back to disk
- The address of the last file block is kept in the file header
- To delete a record, a program must first find its block, copy the block into a buffer, delete the record from the buffer and finally rewrite the block back to the disk
- This leaves unused space in the disk block

Heap files (Unordered Records)

- □ Deleting a large number of records in this way results in wasted storage space
- □ So another technique used for record deletion with a deletion marker stored with each record
- □ To read all records, we create a sorted copy of the file.
- □ Sorting is an expensive operation for a large disk file

Sorted files (Ordered Records)

- We can physically order the records of a file on disk based on their ordering field
- Reading the records from a file is extremely efficient because no sorting is required
- Finding the next record from the current one usually requires no additional block access as next record is in the same block as the current block
- Using a search condition results in faster access when binary search technique is used

Sorted files (Ordered Records)

- Inserting and deleting records are expensive operations for an ordered file because the records must remain physically ordered
- To insert a new record, we must find its correct position in the file and then make space in the file to insert the record in that position
- For a large file this can be very time consuming
- For a deletion, the problem is less severe if deletion marker and periodic reorganization are used
- Ordered files are rarely used in database applications

Hashing Techniques

- Hashing is a primary file organization which provides very fast access to records under certain search conditions
- This organization is usually called a hash file
- The search condition must be an equality condition on a single field called hash field
- Hashing provide a function h, called hash function which is applied to hash field and yields the address of the disk block in which the record is stored

Hashing Techniques

- A search for the record within the block can be carried out in a main memory buffer
- For most records, a single block access is needed to retrieve that record
- Hashing is also used as an internal search structure within a program whenever a group of records is accessed exclusively

Internal hashing

- For internal files, Hashing is implemented as a hash table through the use of an array of records
- The array index range is from 0 to M-1
- Then we have m slots whose address corresponds to array index
- A hash function transforms a hash field value into an integer between 0 to M-1
- A common hash function H(k)=k mod M returns the remainder of an integer hash field K which is used for the record address

Internal hashing

- A collision occurs when the hash field value of a record that is being inserted hashes an address that already contains a different record
- In this situation, we must insert the new record in some other position, since its hash address is occupied
- The process of finding another position is called collision resolution
- A goal a good hashing function is to distribute the records uniformly over the address space so as to minimize collision while not leaving many unused locations

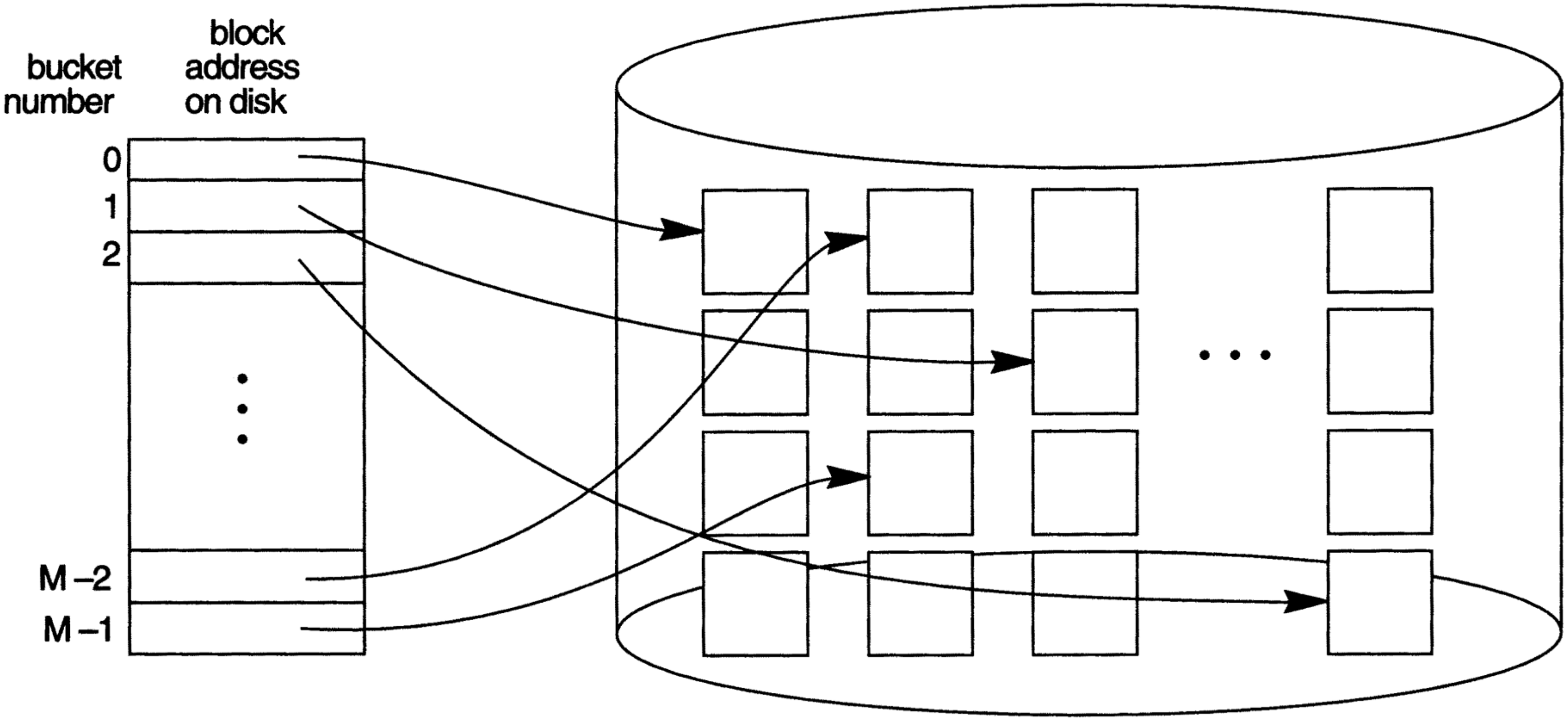
External hashing

- Hashing for disk files is called external hashing
- The target address space is made of buckets, each of which holds multiple records
- A **bucket** in a hash file is unit of storage (typically a disk block) that can hold one or more records.
- A bucket is either one disk block or clusters of contiguous blocks

Static Hashing (external hashing)

- The **hash function**, maps a key into a relative bucket number rather than assigning an absolute block address to the bucket
- A table maintained in the file header converts the bucket number into the corresponding disk block address as shown in diagram
- This hashing scheme is called static hashing because a fixed number of buckets M is allocated

Static Hashing (external hashing)



Matching bucket numbers to disk block addresses

Static Hashing (external hashing)

- It has a serious drawback for dynamic files
- Suppose there are M buckets for the address space and let m be the maximum number of records that can fit in one bucket
- Then at the most (M*m)records will fit in the allocated space
- If number of records are substantially fewer, then left a lot of unused space
- And if number of records increases, collisions will result and retrieval will be slowed down

Dynamic Hashing

- More effective than static hashing when the database grows or shrinks
- **Extendible hashing** splits and combine buckets appropriately with the database size.
  - i.e. buckets are added and deleted on demand.
- Stores an access structure in addition to file
- Access structure is based on the values that result after application of the hash function to the search field

Dynamic Hashing: Advantages & Disadvantages

- Performance of the file does not degrade as the file grows as opposed to static
- no space is allocated in extendible hashing for future growth but additional buckets can be allocated dynamically as needed
- The space overhead for directory table is negligible
- Splitting causes minor reorganization in most cases
- The directory must be searched before accessing the buckets themselves