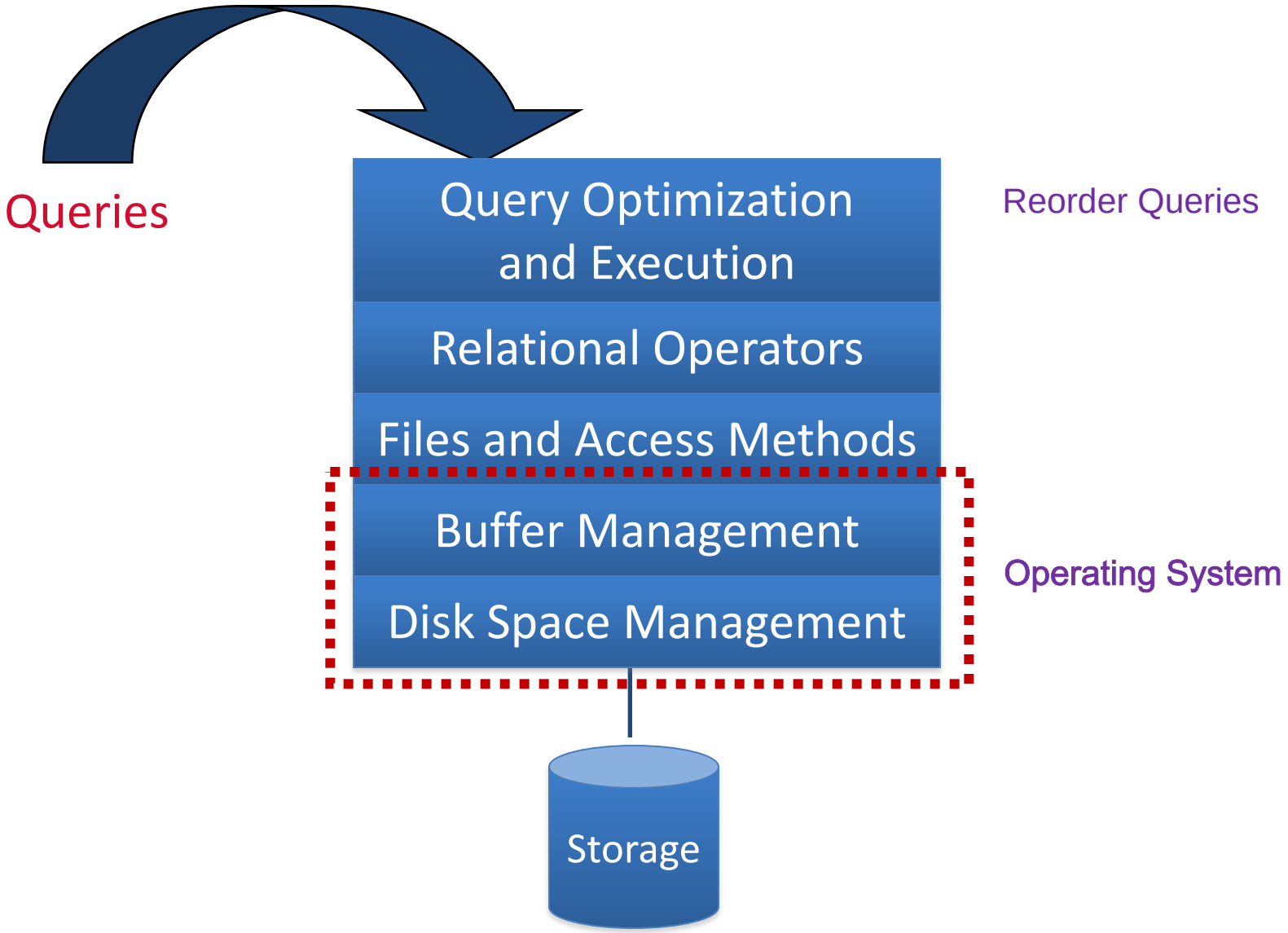
Database Storage Layer

The Storage Layer

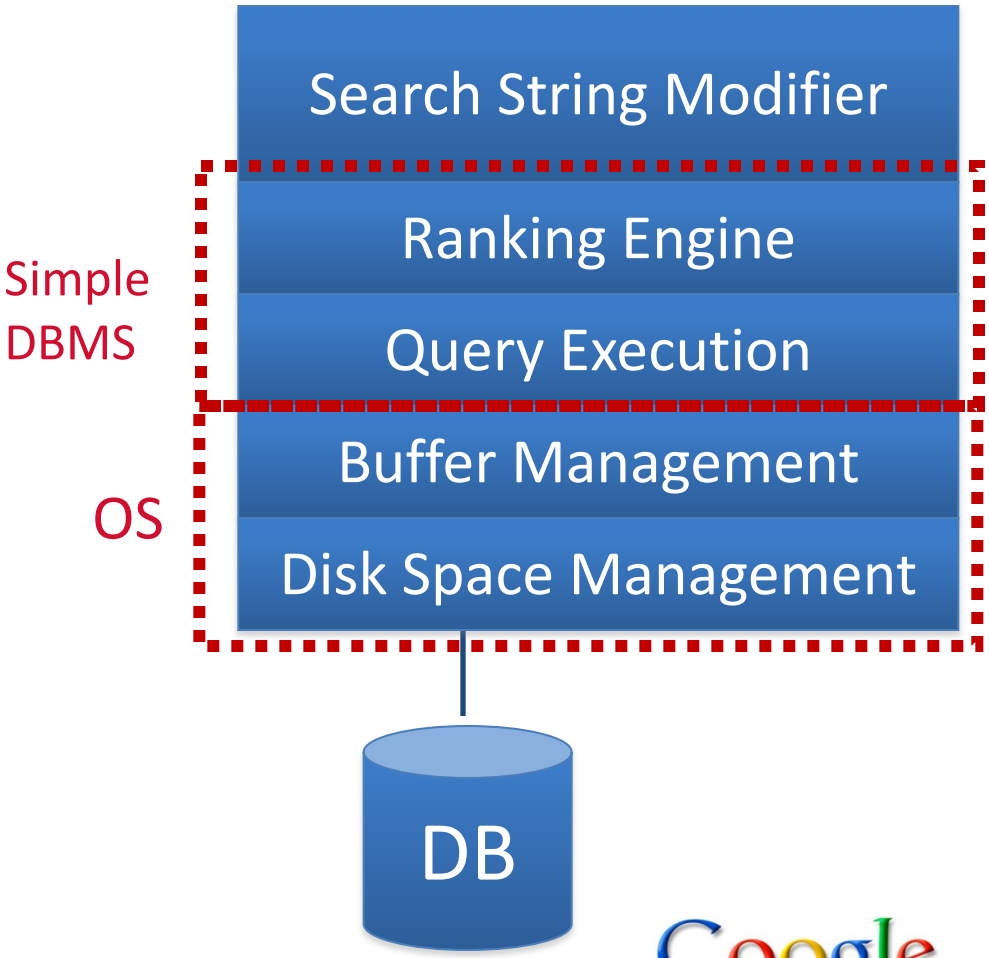
- DBMS layers and storage hierarchy

- Disks

DBMS Layers


Queries
Query Optimization and Execution
Relational Operators
Files and Access Methods
Buffer Management
Disk Space Management
Storage
Reorder Queries
Operating System

A simple search engine



- Simpler system than DBMS
  - Uses OS files for storage
  - One hardwired query
- Typically no concurrency/ recovery
  - Read-mostly, in batches
  - No updates to recover
  - OS a reasonable choice
- Smarts: text tricks
  - Search string modifier (synonyms)
  - Ranking Engine (sort results)
  - no semantics

Google bing

much more complex

Why not OS?

- Layers of abstraction are good … but:
  - Unfortunately, OS often gets in the way of DBMS

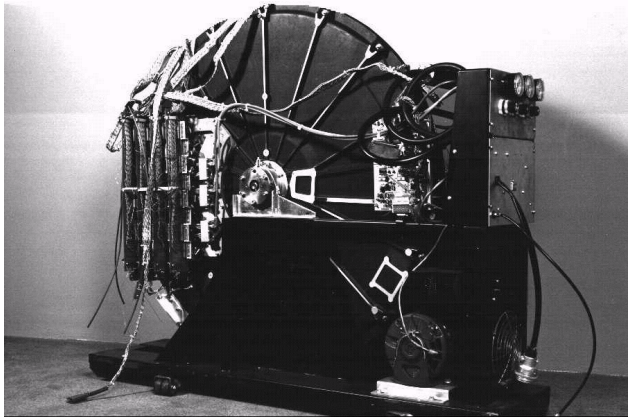
- DBMS needs to do things "its own way"
  - Specialized prefetching
  - Control over buffer replacement policy
    - LRU not always best (sometimes worst!!)
  - Control over thread/process scheduling
    - "Convoy problem"
      - Arises when OS scheduling conflicts with DBMS locking
  - Control over flushing data to disk
    - WAL protocol requires flushing log entries to disk

Least Recently Used

A lock convoy occurs when multiple threads of equal priority contend repeatedly for the same lock

changes are first recorded in the log, which must be written to stable storage, before the changes are written to the database.

Disks and Files



- DBMS stores information on disks.
  - In an electronic world, disks are a mechanical anachronism!
- This has major implications for DBMS design!
  - READ: transfer data from disk to main memory (RAM).
  - WRITE: transfer data from RAM to disk.
  - Both are high-cost operations, relative to in-memory operations, so must be planned carefully!

Why Not Store It All in Main Memory?

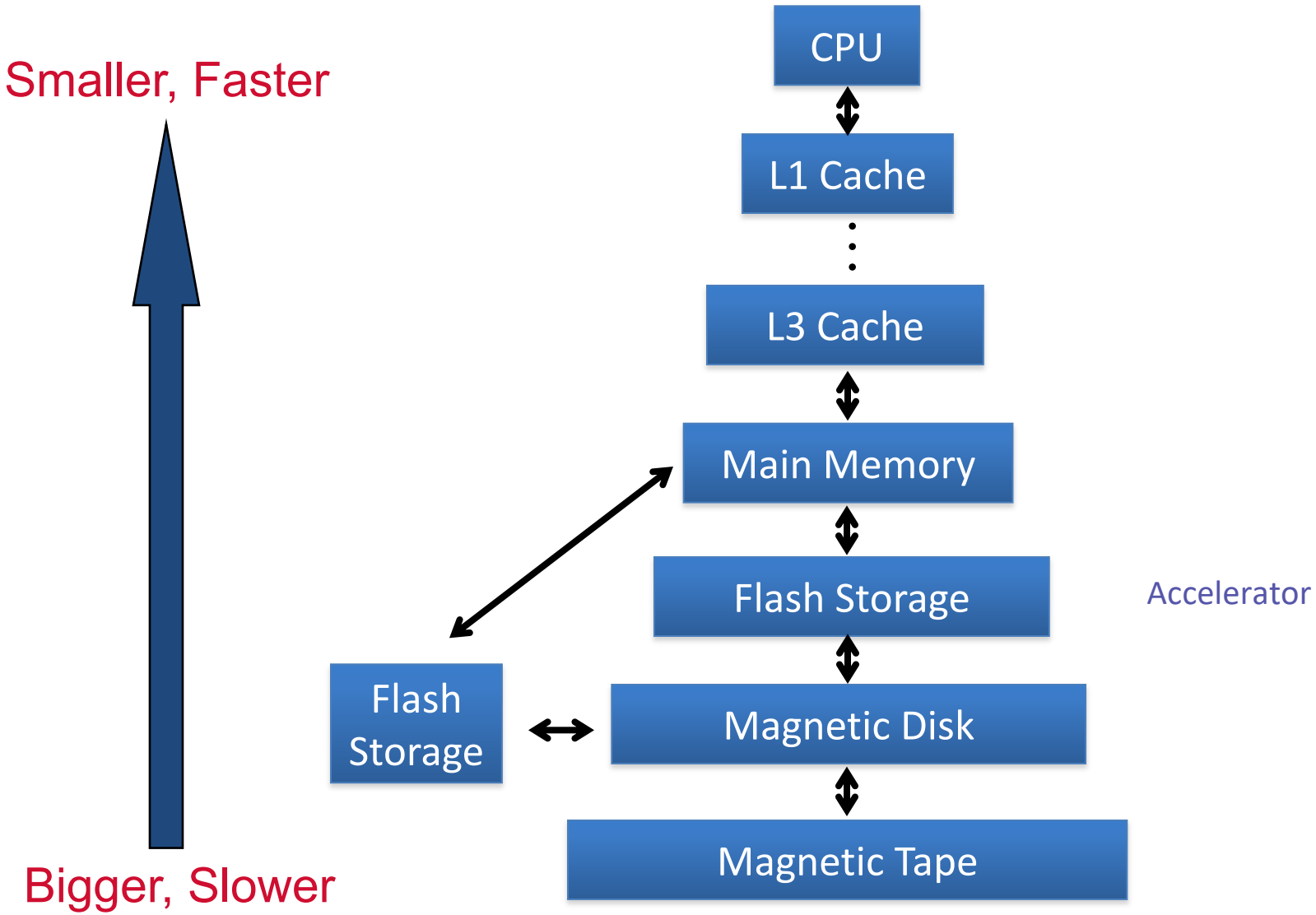
- *Costs too high*
  - High-end Databases today in the Petabyte range.
  - ~ 60% of the cost of a production system is in the disks.
- *Main memory is volatile.* We want data to be saved between runs. (Obviously!)
- *But, main-memory database systems do exist!*
  - Smaller size, performance optimized
  - Volatility is ok for some applications

Redis: in-memory database

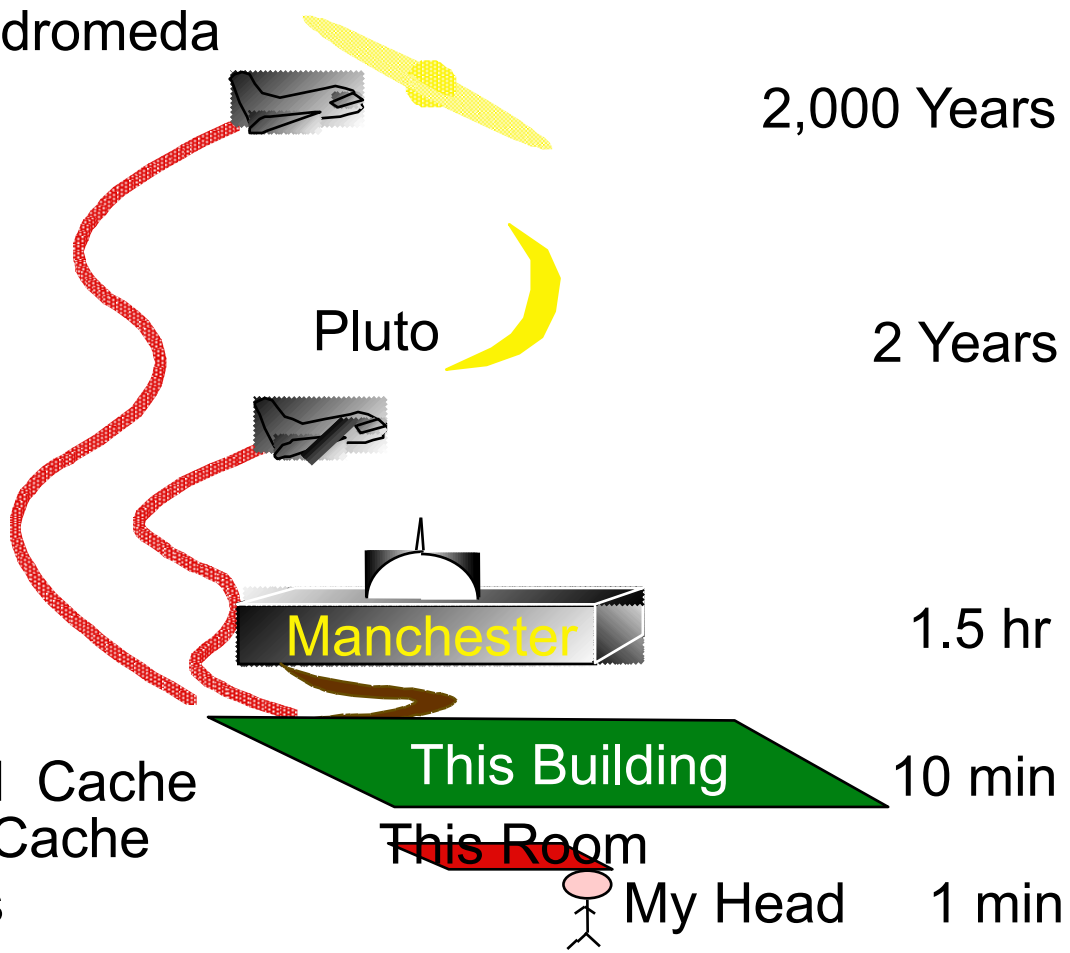
What about Flash?

- Flash chips used for >20 years
- Flash evolved
  - USB keys
  - Storage in mobile devices
  - Consumer and enterprise flash disks (SSD)
- Flash in a DBMS
  - Main storage
  - Accelerator/enabler (specialized cache, logging device)

The Storage Hierarchy



Jim Gray's Storage Latency Analogy: How Far Away is the Data?



10^9	Tape	2,000 Years
10^6	Disk	2 Years
100	Memory	1.5 hr
10	On Board Cache	10 min
2	On Chip Cache	
1	Registers	1 min

The Storage Layer

- DBMS layers and storage hierarchy

- Disks

Disks

- Secondary storage device of choice.
- Main advantage over tapes: *random access* vs. *sequential*.
- Data is stored and retrieved in units called *disk blocks* or *pages*.
- Unlike RAM, time to retrieve a disk page varies depending on location on disk.
  - Therefore, relative placement of pages on disk has major impact on DBMS performance!

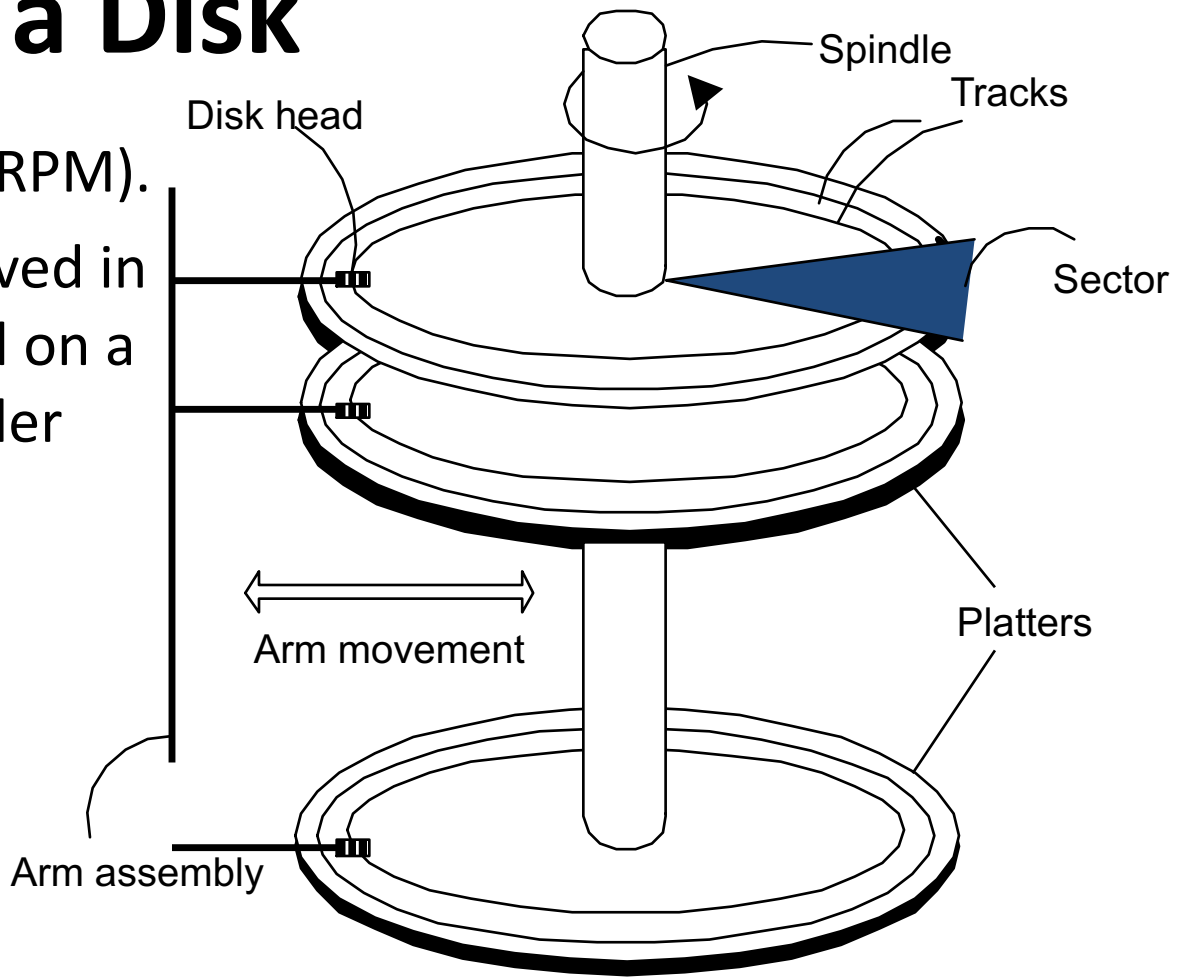
Anatomy of a Disk

The platters spin (5-15 kRPM).

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).
- Newer disks have several "zones", with more data on outer tracks.



13 Outer track can read faster because of density

Each sector can hold 512 bytes of data. A block, on the other hand, is a group of sectors that the operating system can address (point to). A block might be one sector, or it might be several sectors (2,4,8, or even 16). The bigger the drive, the more sectors that a block will hold.

Accessing a Disk Page

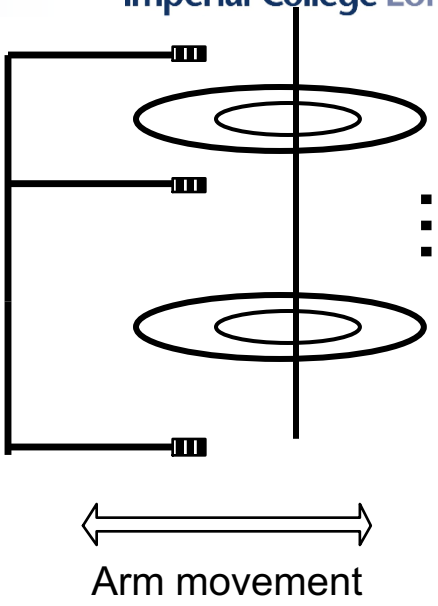
Time to access (read/write) a disk block:

- *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)

Settle time is how long the head assembly takes to fully lock on to a track and stop oscillating, so it can begin reading. It would form the last part of the entire seek time, in the stage where the heads decelerate and come to a rest.
Rotation delay is how long it takes the platters to rotate the required sector into position underneath the head.

Seek Time

The actual physical positioning of the read/ write head of the disc is called seeking. The amount of time that it takes the read/write head of the disc to to move from one part of the disk to another is called the seek time.



Head positioning

3x to 20

Time

x

1

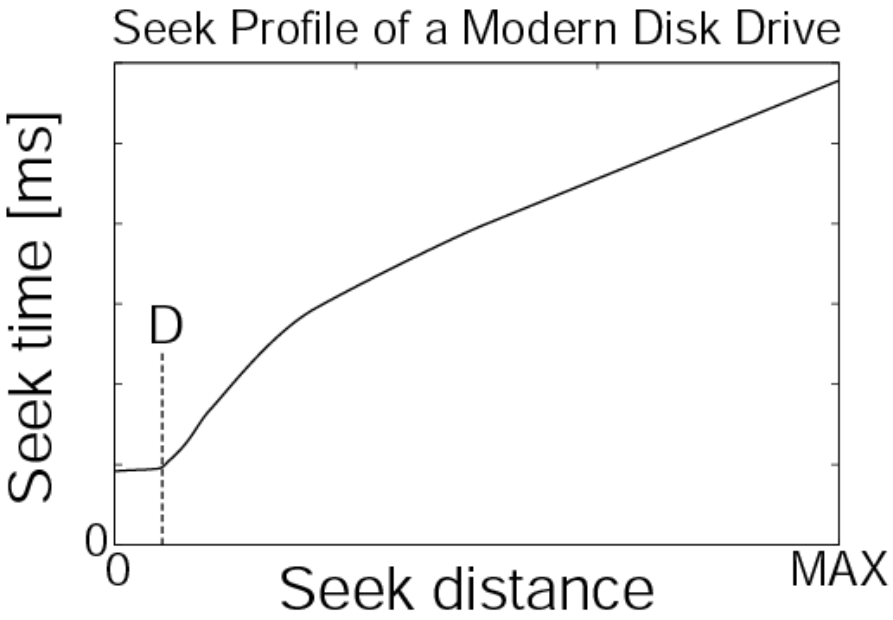
N

Cylinders Traveled

For small cylinders, the only need is to reposition the head instead of the arm movement. SO the seek time of small # cylinders seek time is dominated by head position

Seeking in Modern Disks

Seek time discontinuity

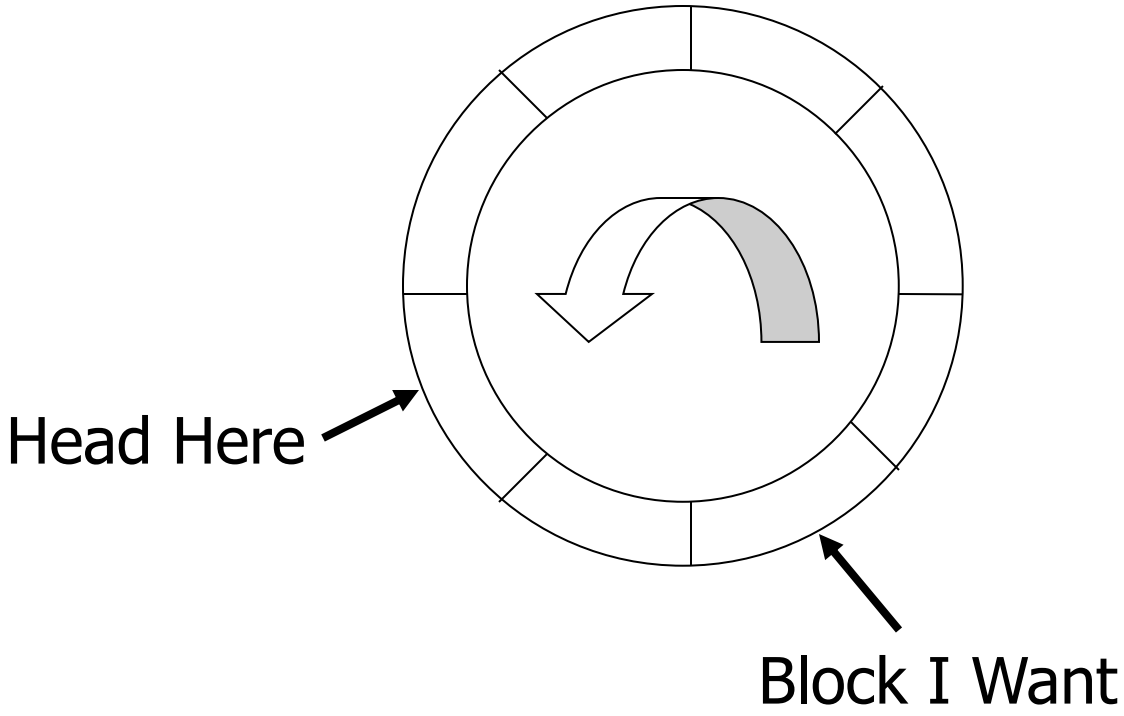


Seek time -- the time for the head to move from its current cylinder to the targer cylinder.
Settling time -- the time to position the head over the target track until the correct track identification is confirmed. A typical seek time is 10 ms.

Short seeks are dominated by "settle time"

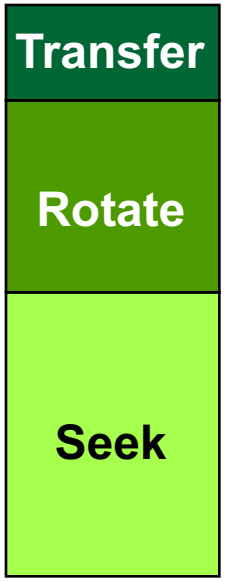
- Move to one of many nearby tracks within settle time
- D is on the order of tens to hundreds
- D gets larger with increase of disk track density

Rotational Delay



Seek Time & Rotational Delay Dominate

- Seek time varies from about 1 to 20 ms
- Rotational delay varies from 0 to 10 ms
- Transfer rate is < 1ms per 4KB page



- Key to lower I/O cost: reduce seek/rotation delays!

- Also note: For shared disks most time spent waiting in queue for access to arm/controller

Arranging Pages on Disk

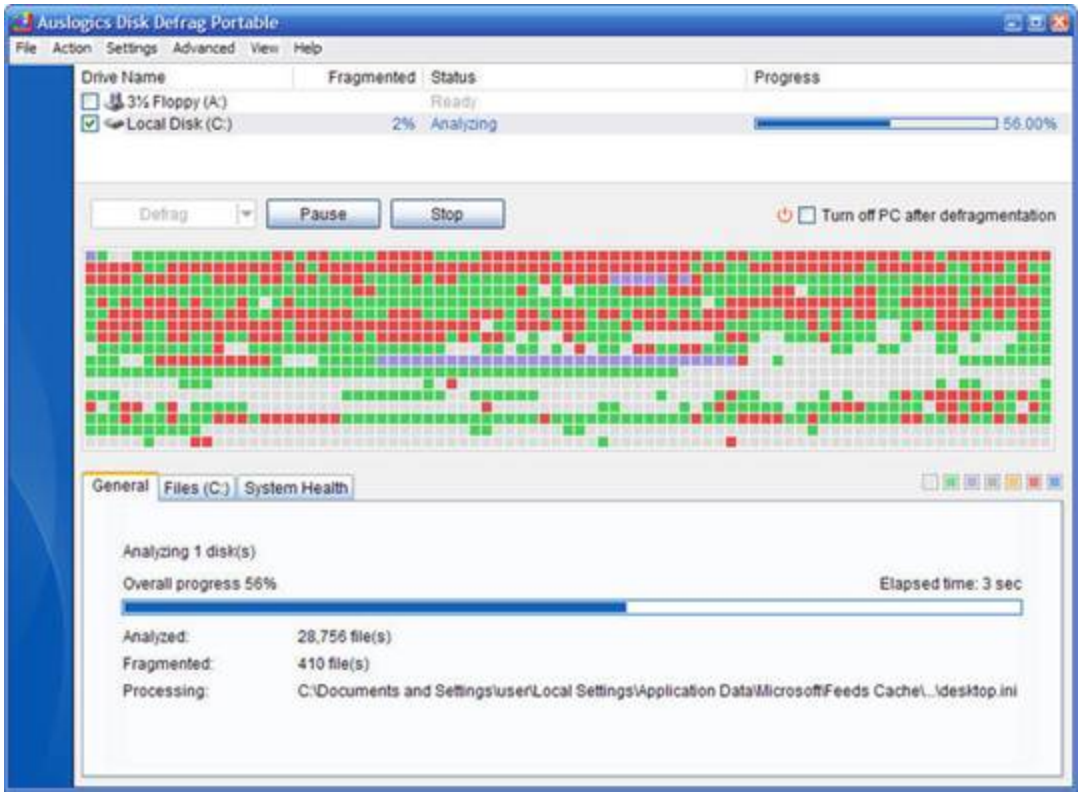
- *"Next"* block concept:
  - blocks on same track, followed by
  - blocks on same cylinder, followed by
  - blocks on adjacent cylinder

Cannot control the actual space it is written in.

- Blocks in a file should be arranged sequentially on disk (by "next") to minimize seek and rotational delay.

- An important optimization: pre-fetching

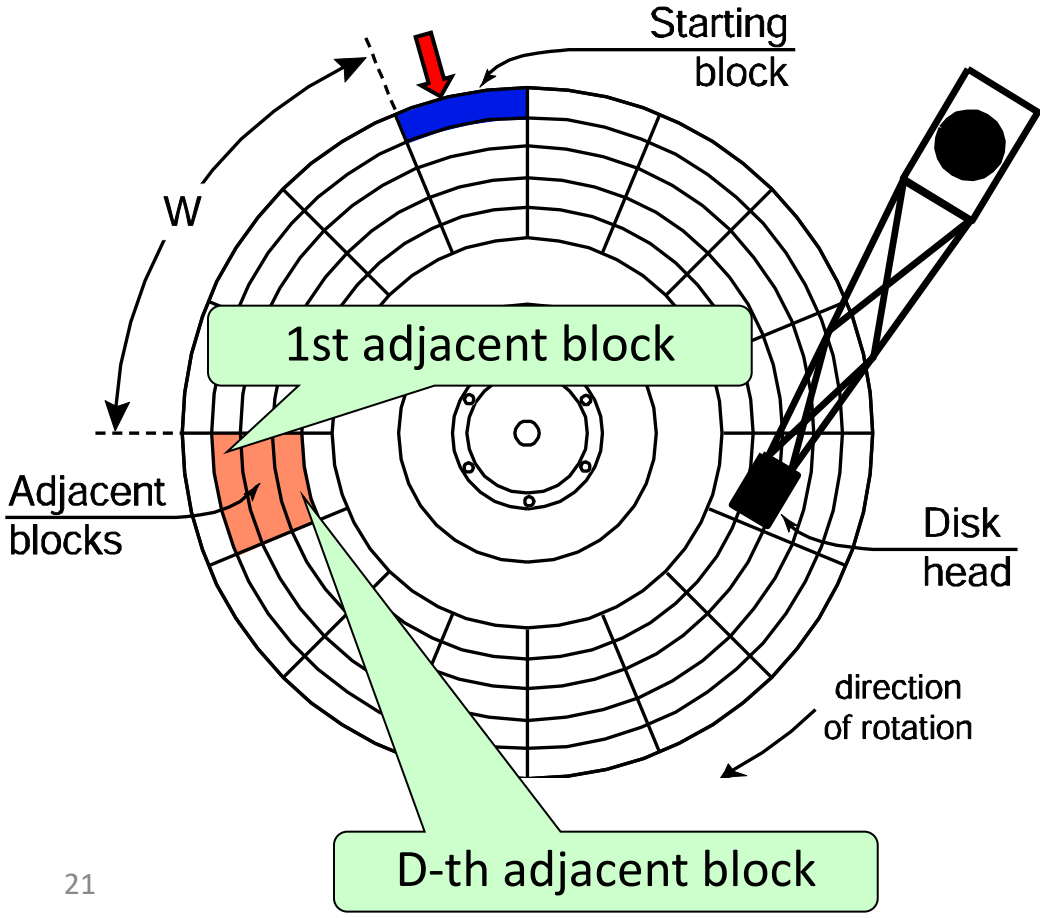
Remember Defrag



Define adjacent blocks

- Access incurs settle time only
- Equidistant wrt access time from starting block

Adjacent means the "next" block that will be read under RPM. Not the actual physical adjacent.



D: # of adjacent blocks

W: degree disk will rotate during settle time

Disk block has more than one neighbor

Rules of thumb…

1. Memory access much faster than disk I/O (~ 1000x)
2. “Sequential” I/O faster than “random” I/O (~ 10x)

Disk Space Management

- Lowest layer of DBMS software manages space on disk
- Higher levels call upon this layer to:
  - allocate/de-allocate a page
  - read/write a page
- Best if a request for a *sequence* of pages is satisfied by pages stored sequentially on disk! Higher levels don't need to know if/how this is done, or how free space is managed.

Summary

Key to store data on disk is:

1. Store data together if it is queried together
2. Avoid random access and use sequential access where possible: use cost model for it
3. Unit to optimize for is disk page: align data structures for page size