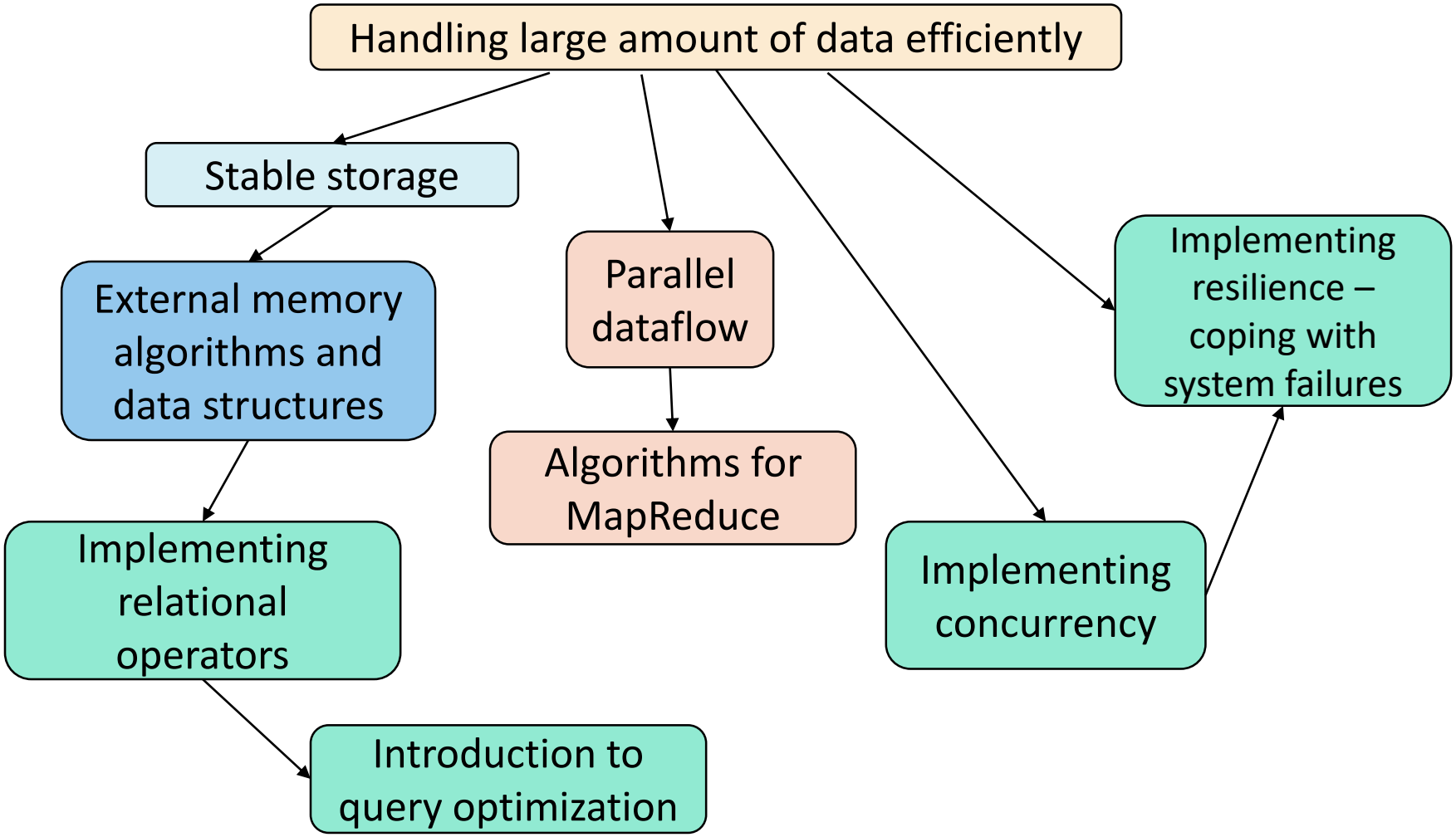
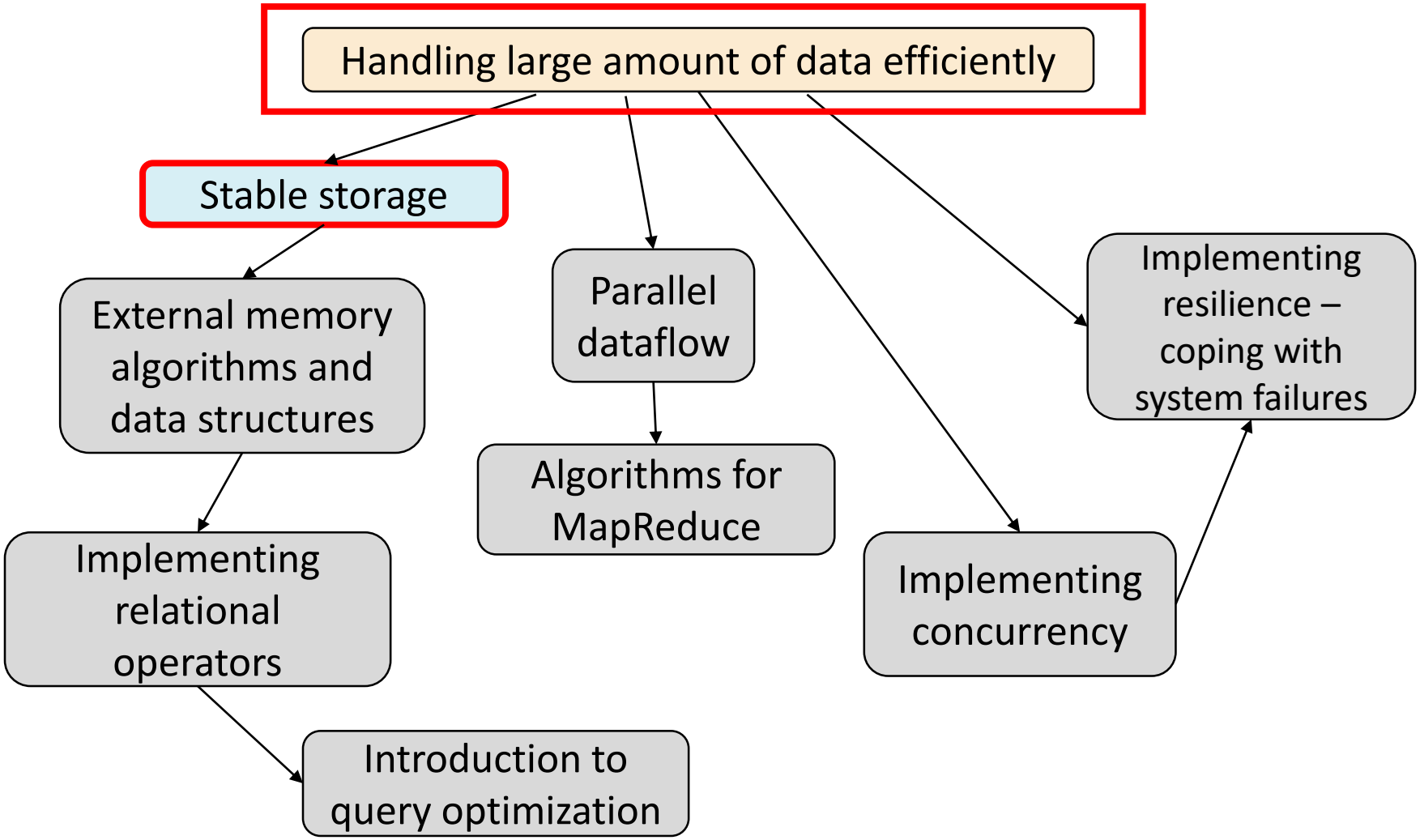
Roadmap


Handling large amount of data efficiently
Stable storage
External memory algorithms and data structures
Parallel dataflow
Algorithms for MapReduce
Implementing relational operators
Introduction to query optimization
Implementing concurrency
Implementing resilience – coping with system failures

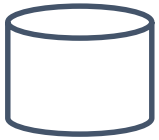
Setting up the stage: storage media



Lecture 01.01

Physical storage media and its constraints

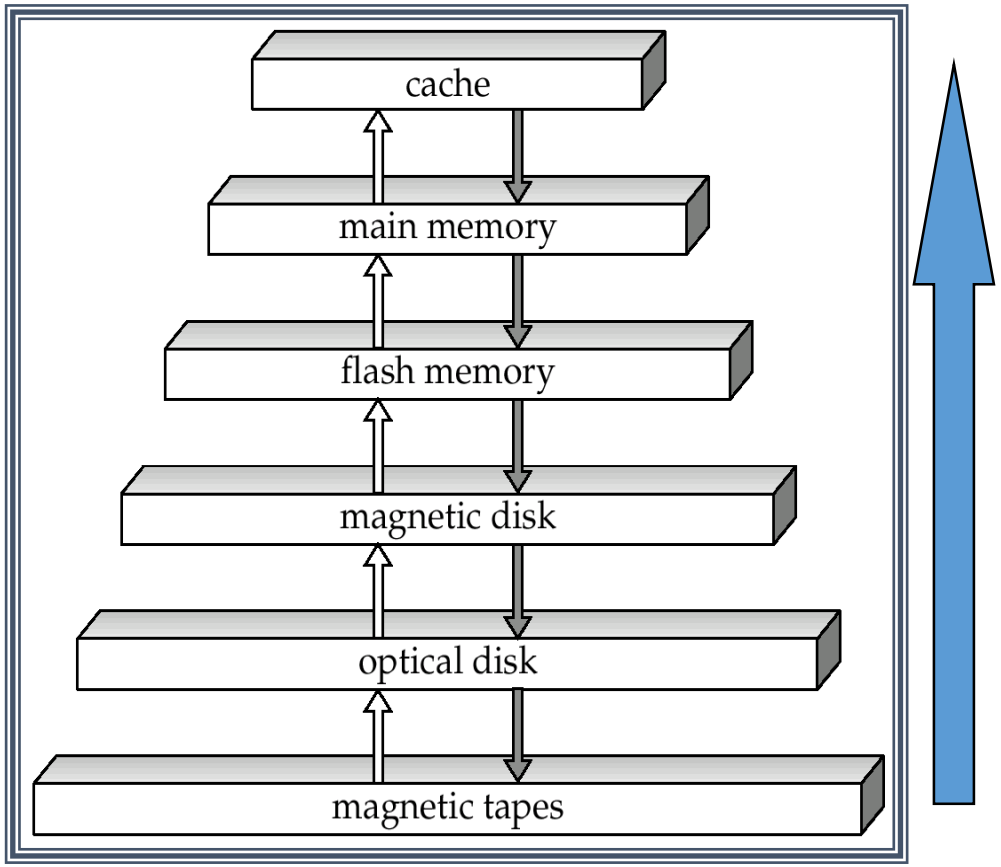
Where to store all this big data



By Marina Barsky
Winter 2017, University of Toronto

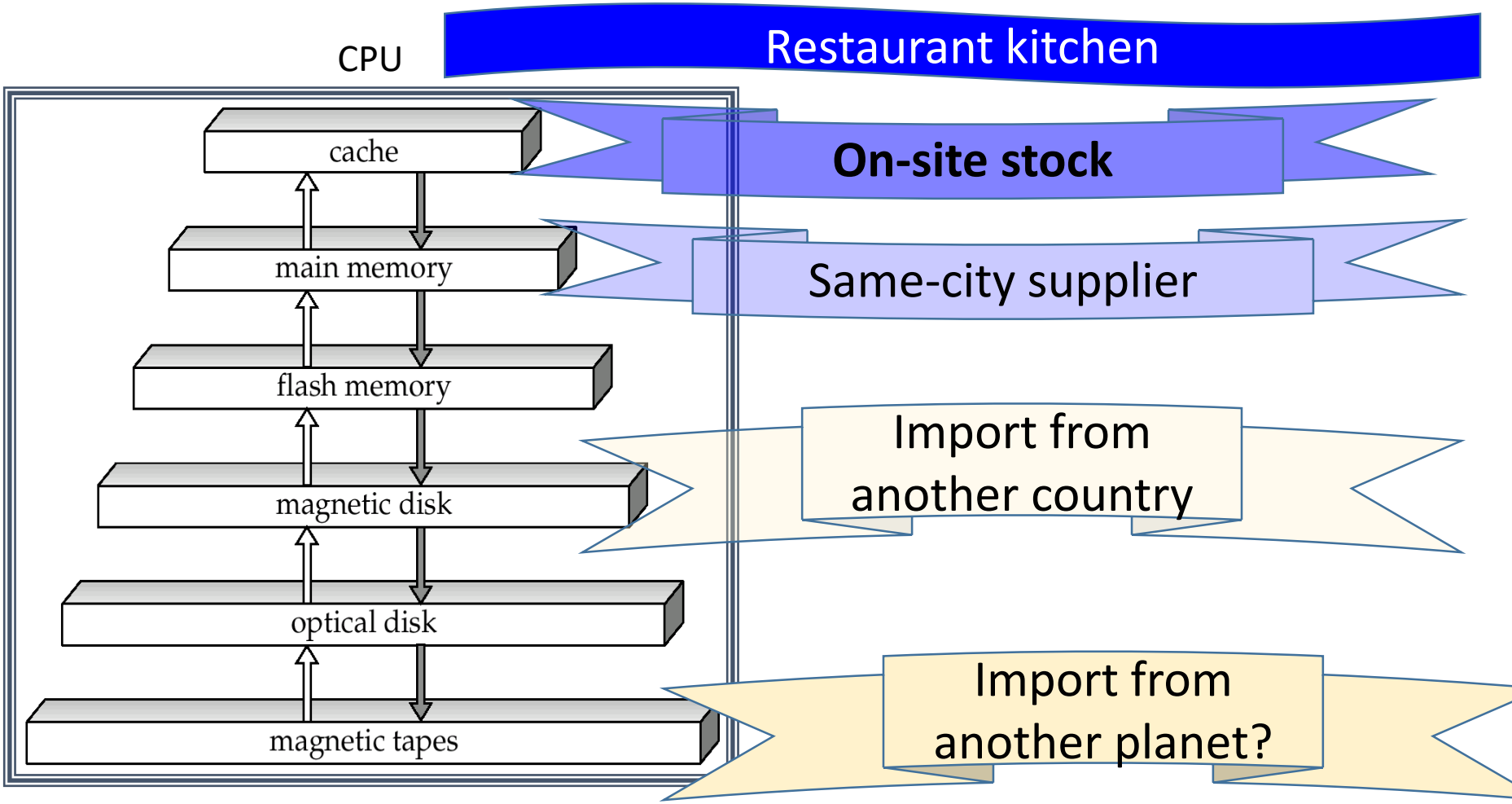
The memory hierarchy

Smaller, Expensive



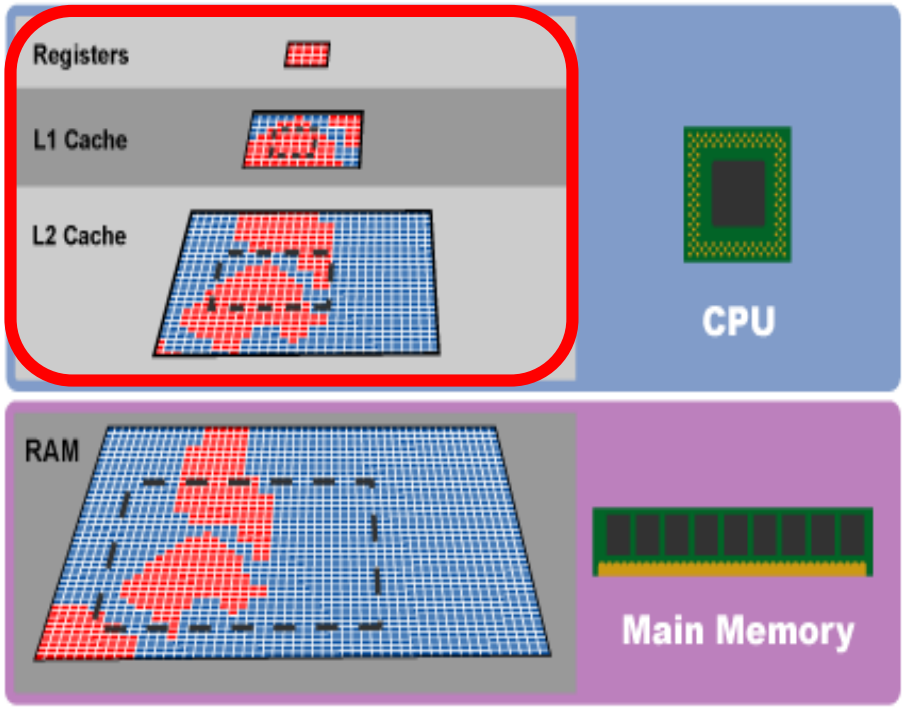
Bigger, Cheaper

The memory hierarchy analogy: downtown restaurant



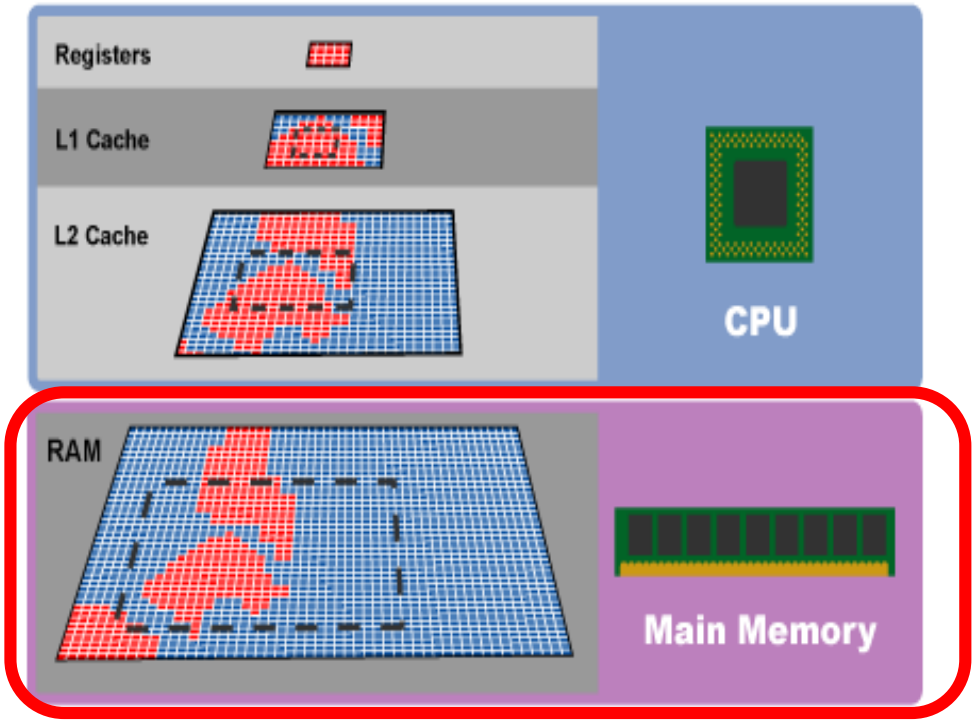
Adopted from Jon Stokes: http://arstechnica.com/gadgets/2002/07/caching/2/

Cache



- Data access rate: 40-**600GB**/sec)
- Expensive to implement (4-6 transistors per cell)
- Typically on chip (Close to CPU)
- Cache misses are expensive
  - Similar tradeoffs as between main memory and disks
- Cache-coherency for multiple CPUs (concurrency ?)

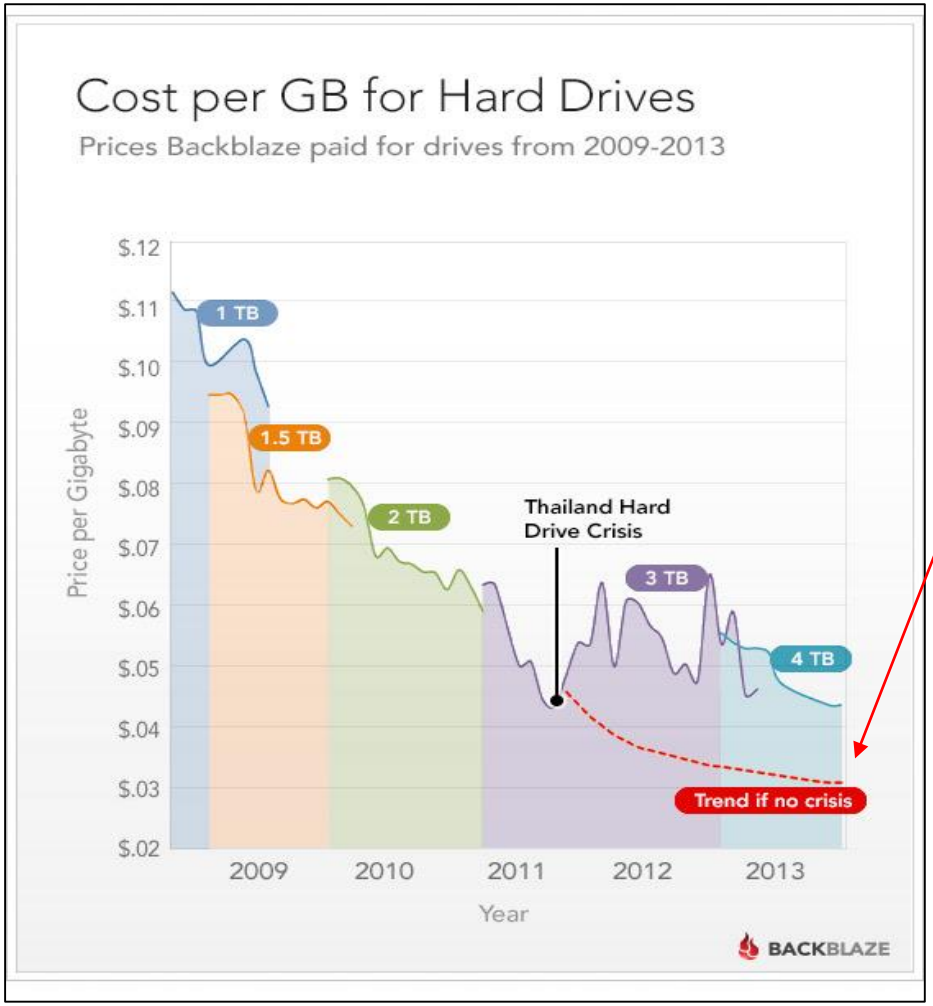
Main memory



- Still fast (**10GB** /sec)
- Volatile: requires refresh to keep the charge
- Directly connected to the CPU through memory bus
- Cheap and dropping: 1GB < 100$
  - Main memory databases feasible now-a-days

http://www.cdf.toronto.edu/~csc443h/winter/volatilememory.html

Magnetic disks



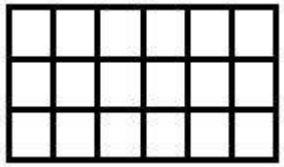
- Non-volatile storage based on magnetized-demagnetized areas
- Very cheap
- Allows sequential and random access
- Random access is slow:

100MB/sec transfer rate

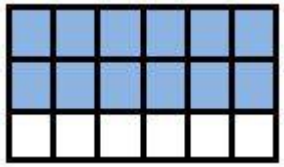
Flash memory

- Electrically erasable programmable read-only memory: floating gate transistors
- Non-volatile, but has a limited number of write-erase cycles
- Fast for reading (**700MB**/sec), slow for writing especially overwriting – applies high voltage
- Proposed as an intermediate buffer between disk and RAM for database systems

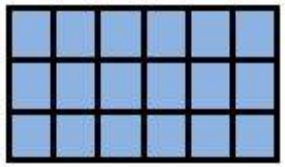
(see paper by G. Graefe)



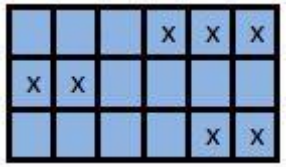
empty block



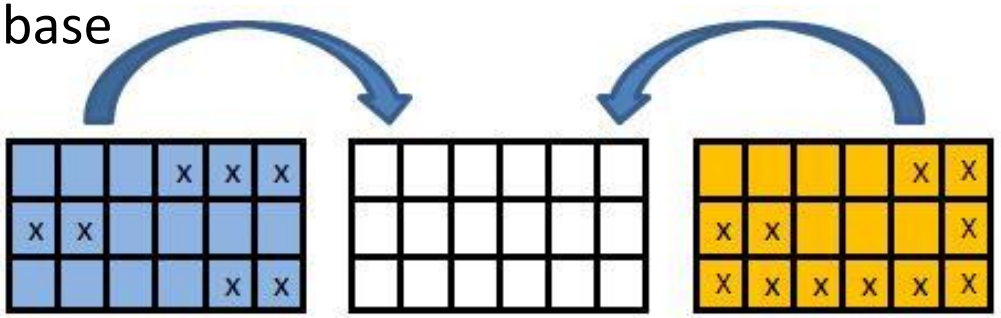
data written



block is full



mark for deletion



"Good data" is rewritten into an empty block

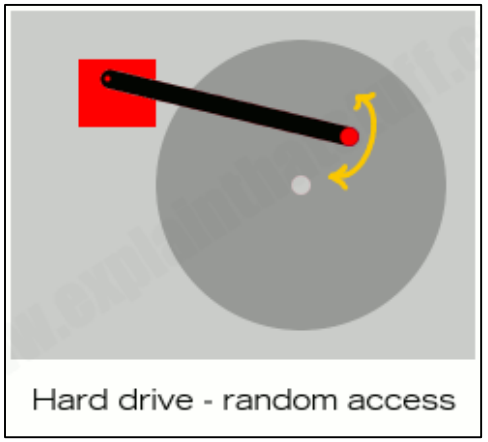
Optical disks

- CDs/DVDs; Jukeboxes
- Using laser beam to read and to write data
- Used more as backups
- Low capacity
- Very slow to write (if possible at all). Limited number of write-erase cycles.
- Less reliable

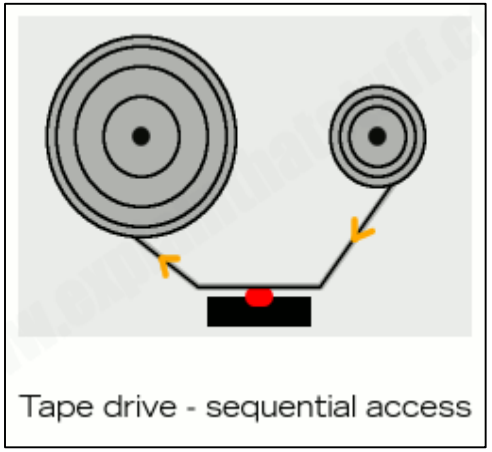
An exploded disk

Magnetic tapes

- Backups; super-cheap; painful to access – sequential access only
- Not suitable for operational databases
- IBM just released a secure tape drive storage solution

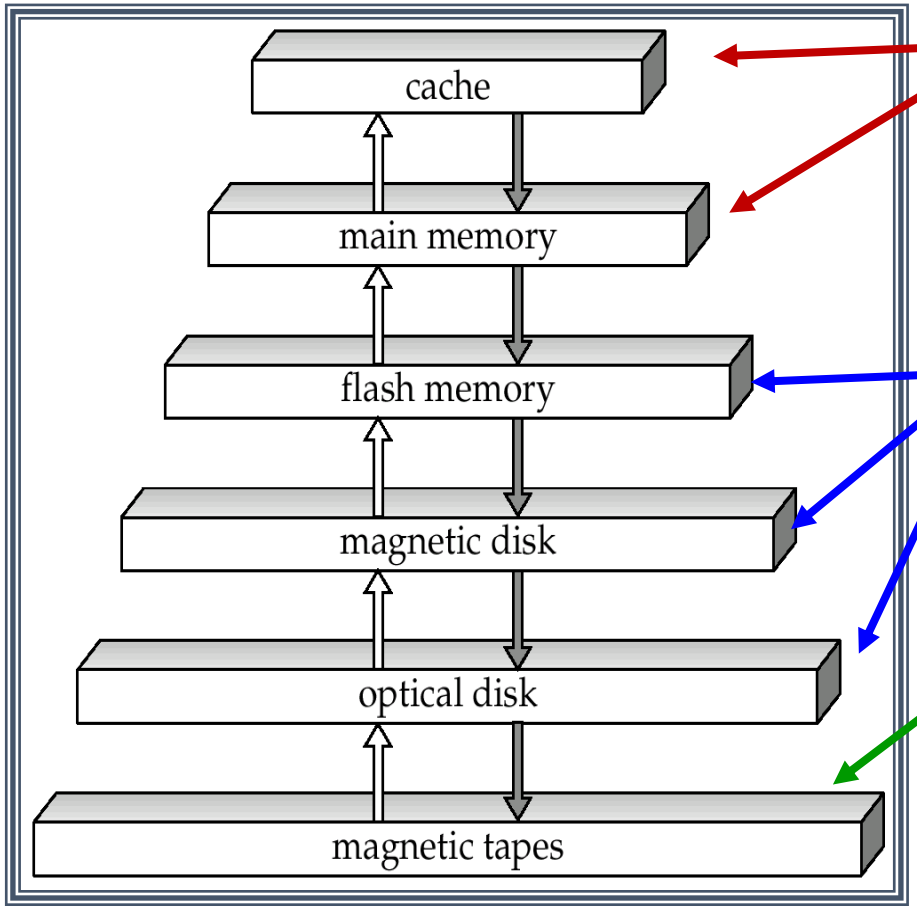


Hard drive - random access



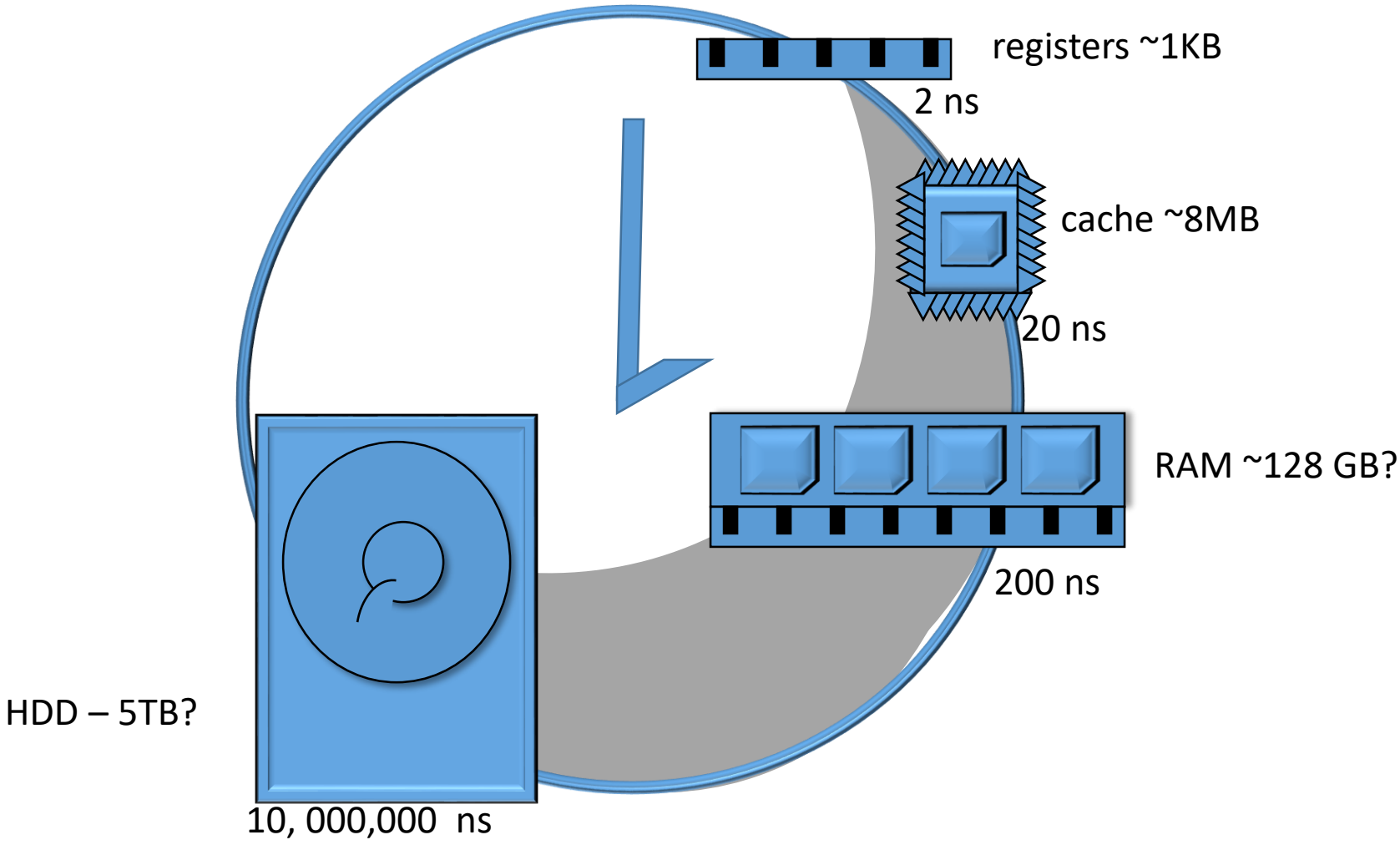
Tape drive - sequential access

Storage is divided into:



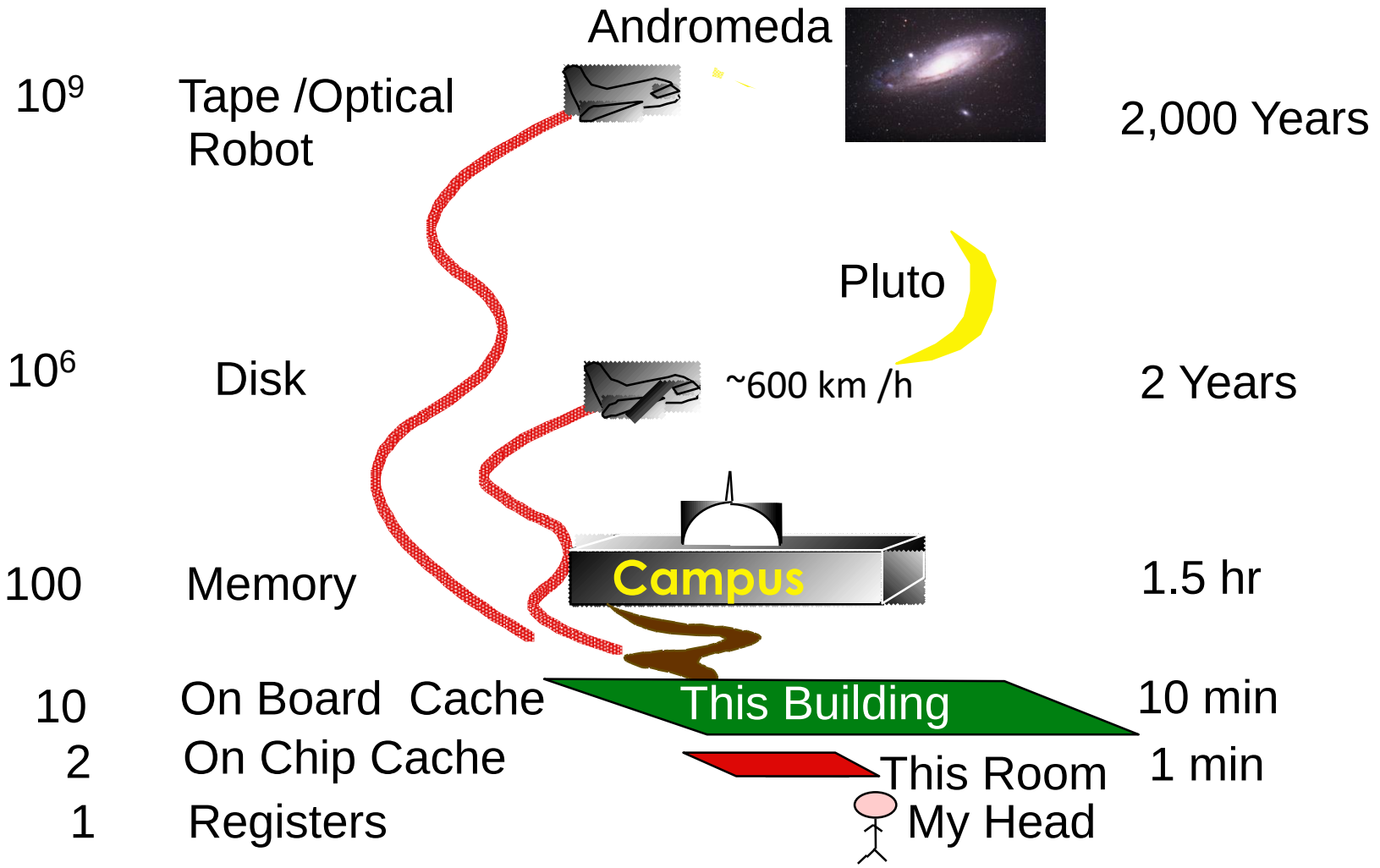
- **Primary**
  volatile, fast, expensive
- **Secondary**
  non-volatile, cheap
- **Tertiary**
  non-volatile, super cheap, slow

Do we really have a choice of storage media?



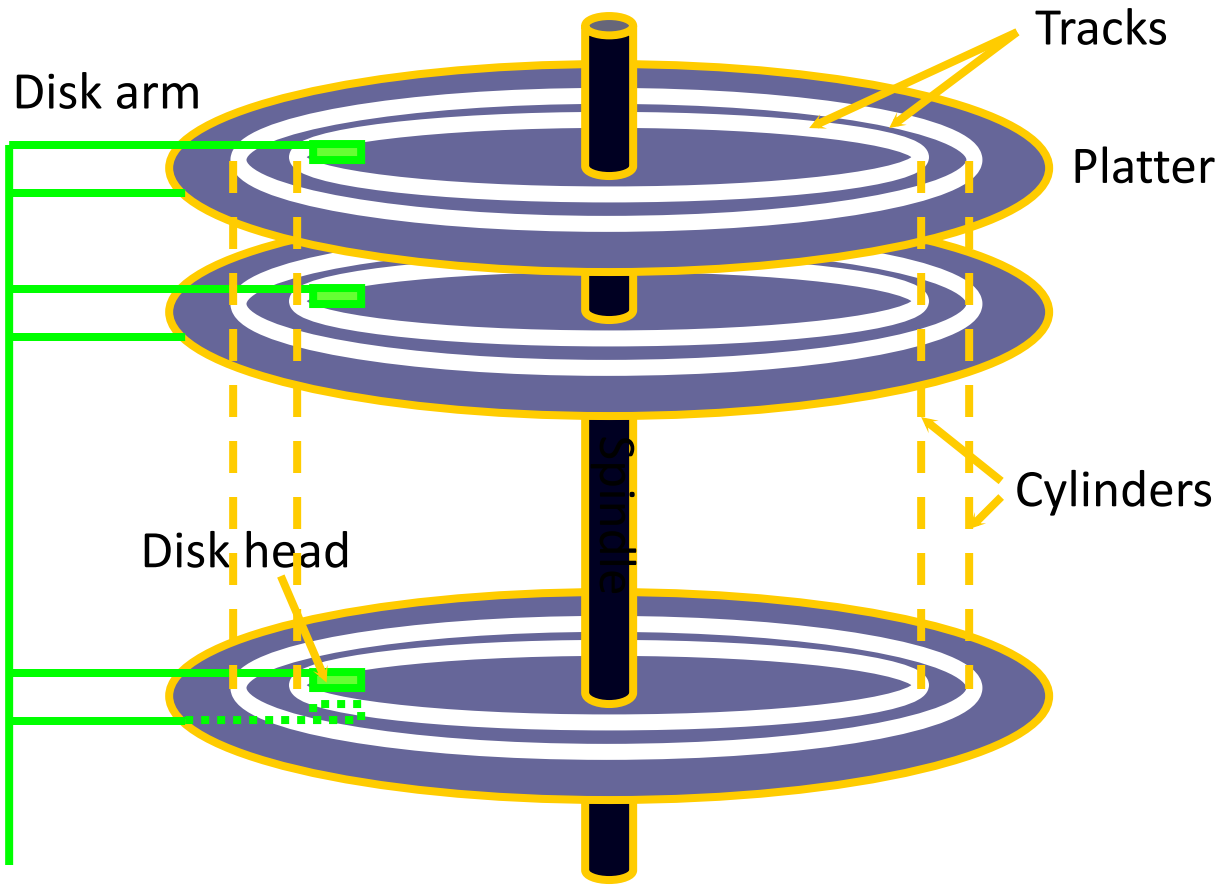
1 ns - speed-of-light (a photon) travel a 1 ft (30.5cm) distance

Storage Latency:
How Far Away is the Data?



By Jim Gray

Disk anatomy



Arm movement

Spindle rotation

“Moving parts” are slow

Disk platter

- Surfaces (1 or 2) of a platter are covered with concentric ***tracks***.
  - Tracks at a common radius = ***cylinder***.
  - All data of a cylinder can be read quickly, without moving the heads.
  - Typical big disk: 2^{16} = 65,536 cylinders.

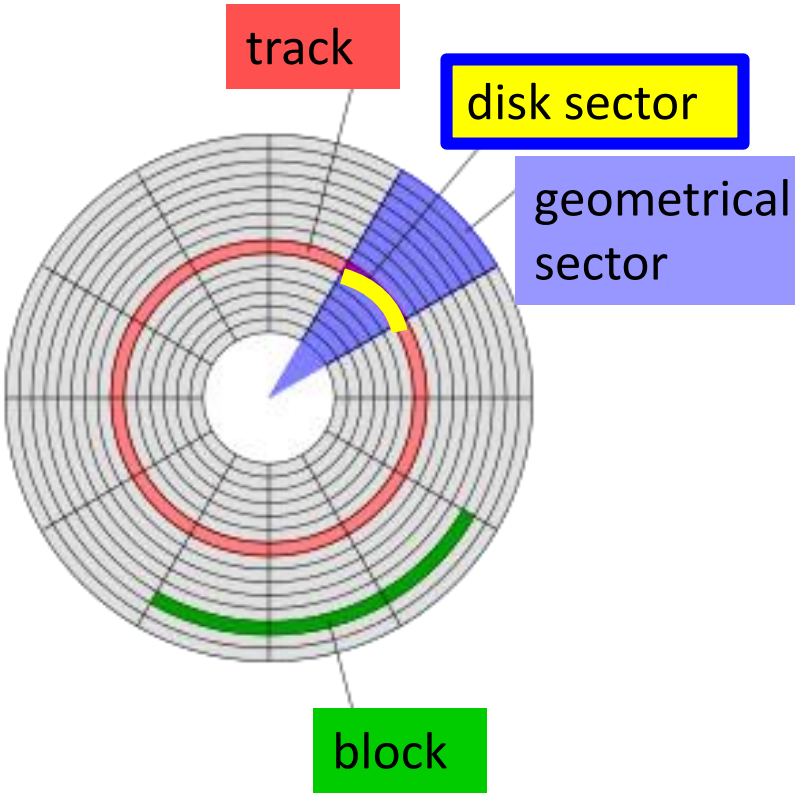
```
$ sudo hdparm -I /dev/sda

/dev/sda:
ATA device, with non-removable media
        Model Number:       ST3500630AS
        Serial Number:      9XXYZ845YZ
        Firmware Revision:  3.AAK
Standards:
        Supported: 7 6 5 4
        Likely used: 7
Configuration:
        Logical         max     current
        cylinders       16383   16383
        heads           16      16
        sectors/track   63      63
        --
        CHS current addressable sectors:
 …
```

Sector

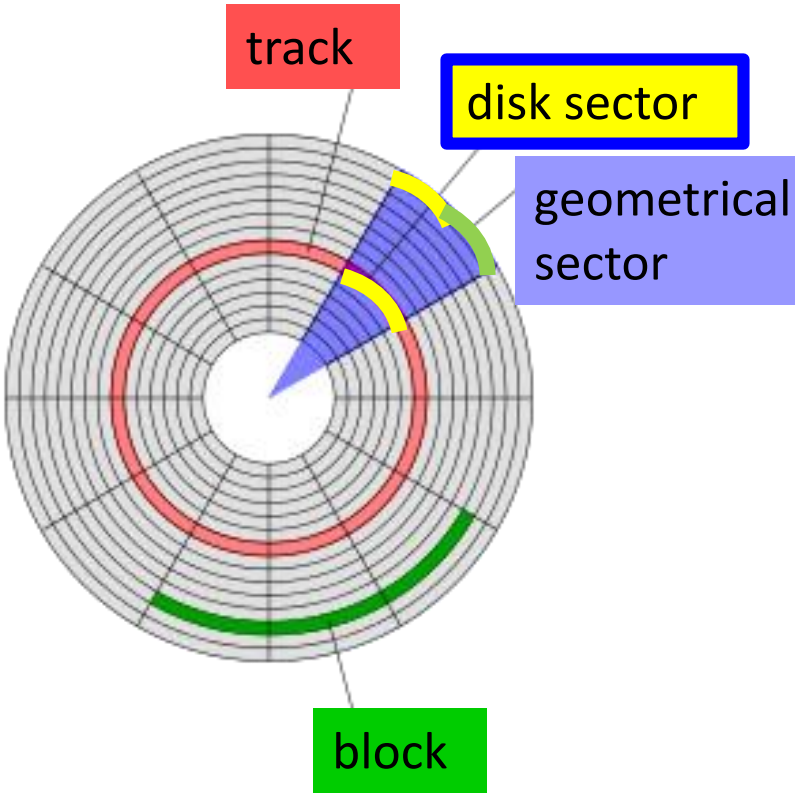
- Tracks are divided into ***sectors*** by unmagnetized gaps (which are 10% of track).
- Typical track: 256 sectors.

- ***Sector*** – the minimum physical storage unit of a hard drive. Stores a fixed amount of data (traditionally 512 bytes, now 4096 bytes). The size is associated with the particular disk and **cannot be changed**.



Bit density

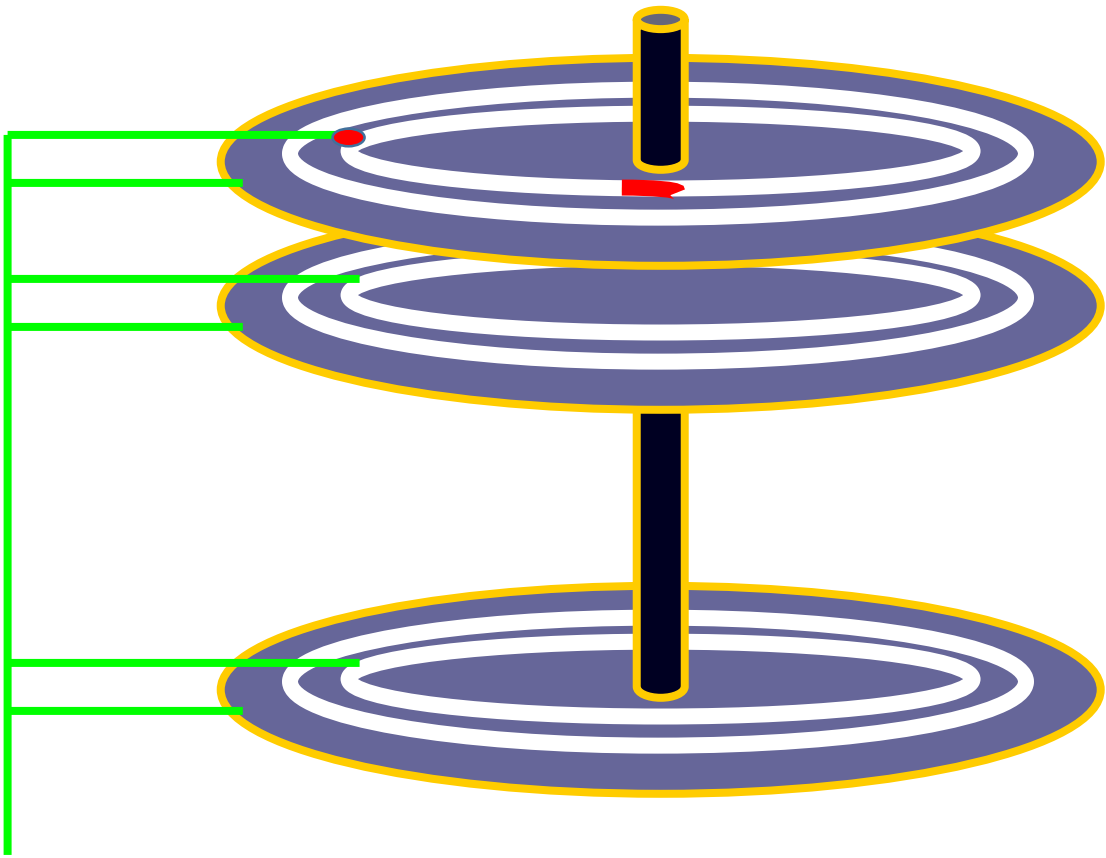
- The sectors towards the outside are physically longer. Each sector contains the same number of bytes, the outer have lower *bit density*.
- Current solution - *zone bit recording*: tracks are in different zones. Each zone divided into sectors of a similar physical size.
- Outer zones have more sectors → contiguous reads and writes are faster on outer tracks, as more bits pass under the head with each rotation; (~25% difference).



How disks work

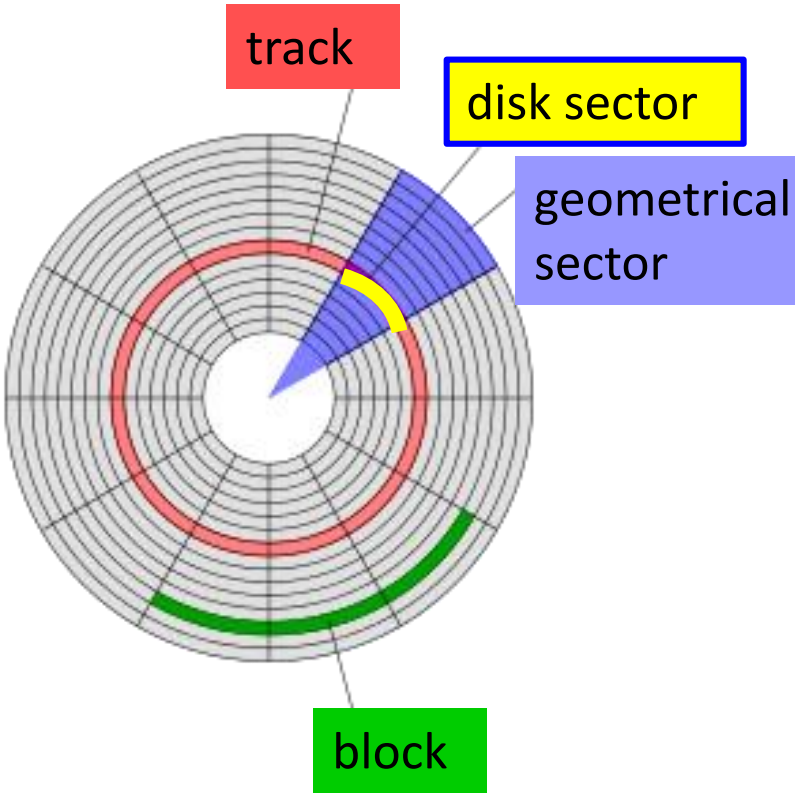
https://www.youtube.com/watch?v=H8VUB7k-4Ls

- To access data: head moves to the desired track and waits until the desired sector comes underneath
- Bit on/off corresponds to the direction of a magnetic field
- The disk head does not touch the platter. If disk is dropped, the head can come to a contact with the platter – ***disk crash*** – irreparable damage



Blocks

- The ***Block*** (transferred as a ***Page*** to RAM) is a fixed-size portion of secondary storage corresponding to the amount of data transferred in a single access and physically occupies one or more consecutive sectors
  - Typical block size (Oracle): 1, 4, 8, 16, or 32 KB.
  - Has to be set before creating database
- **The data is read and written in blocks**



Moor's law and memory latencies

Doubles every 18 months:

- The speed of processors, i.e., the number of instructions executed per second and the ratio of the speed to cost
- The cost of main memory per bit and the number of bits that can be put on one chip
- The cost of disk per bit and the capacity of the largest disks

Exceptions:

- The speed of accessing data in main memory.
- The speed at which disks rotate.

Latency becomes **progressively larger**: the time to move data between levels of the memory hierarchy appears to take progressively longer compared with the time to compute

Disk latency (transfer time)

Sum of:

- Seek time: time to move heads to proper cylinder
- Rotational delay: time for desired block to come under head
- Transfer time: time to read/write data in the block (= time for disk to rotate over the block)

Random disk access

Seek time + rotational delay + transfer time

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

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

Sequential disk access improves performance

Seek time + rotational delay + transfer time

- Seek time
  - 0 (assuming data is on the same track)
- Rotational delay
  - 0 (assuming data is in the next block on the track)

Easily an order of magnitude faster than random disk access!

Performance tricks

1. Disk layout strategy
   - Keep related things (what are they?) close together: same sector/block ! same track ! same cylinder ! adjacent cylinder
2. Double buffering
   - While processing the current block in memory, prefetch the next block from disk (overlap I/O with processing)
3. Track buffer
   - Read/write one entire track at a time
4. Parallel I/O
   - More disk heads working at the same time (too complex to implement)

Quick question

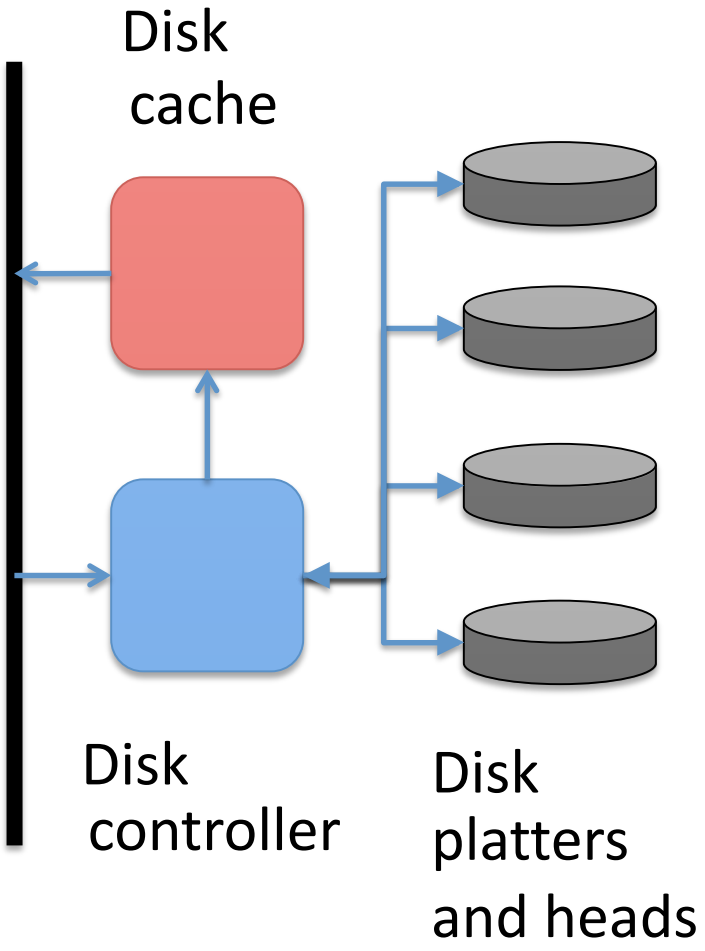
If a location of the block inside the platter is defined by the radius of the cylinder and the sector number, and each block occupies 2 sectors,

where would the storage manager write the block to be the closest to the (radius=2, sector=4)?

A. radius=3, sector=4
B. radius=3, sector=6
C. radius=2, sector=6
D. radius=2, sector=7

Disk Controllers

1. Schedule the disk heads
2. Buffer data in and out of disk - prefetching
3. Manage the "bad blocks" so they are not used.

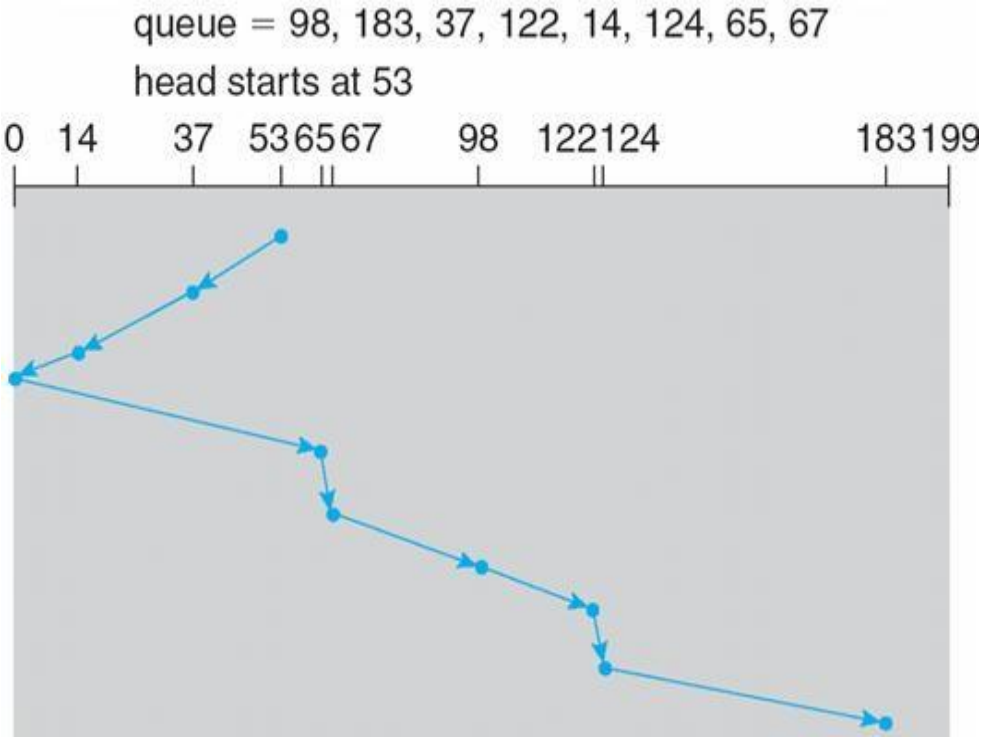


Disk controller – scheduling

Algorithm is named after the behavior of an elevator:

- the elevator continues to travel in its current direction (up or down) until empty, stopping only to let individuals off or to pick up new individuals heading in the same direction.

SCAN (elevator algorithm)



Algorithms for external memory

- In most studies of algorithms, one assumes the "RAM model":
  - Data is in main memory,
  - Access to any item of data takes as much time as any other.
- When implementing a DBMS, one must assume that the data does ***not*** fit into main memory.
- Often, the best algorithms for processing very large amounts of data differ from the best main-memory algorithms for the same problem.
- There is a great advantage in choosing an algorithm that uses **few disk accesses**, even if the algorithm is not very efficient when viewed as a main-memory algorithm.

I/O model of computation

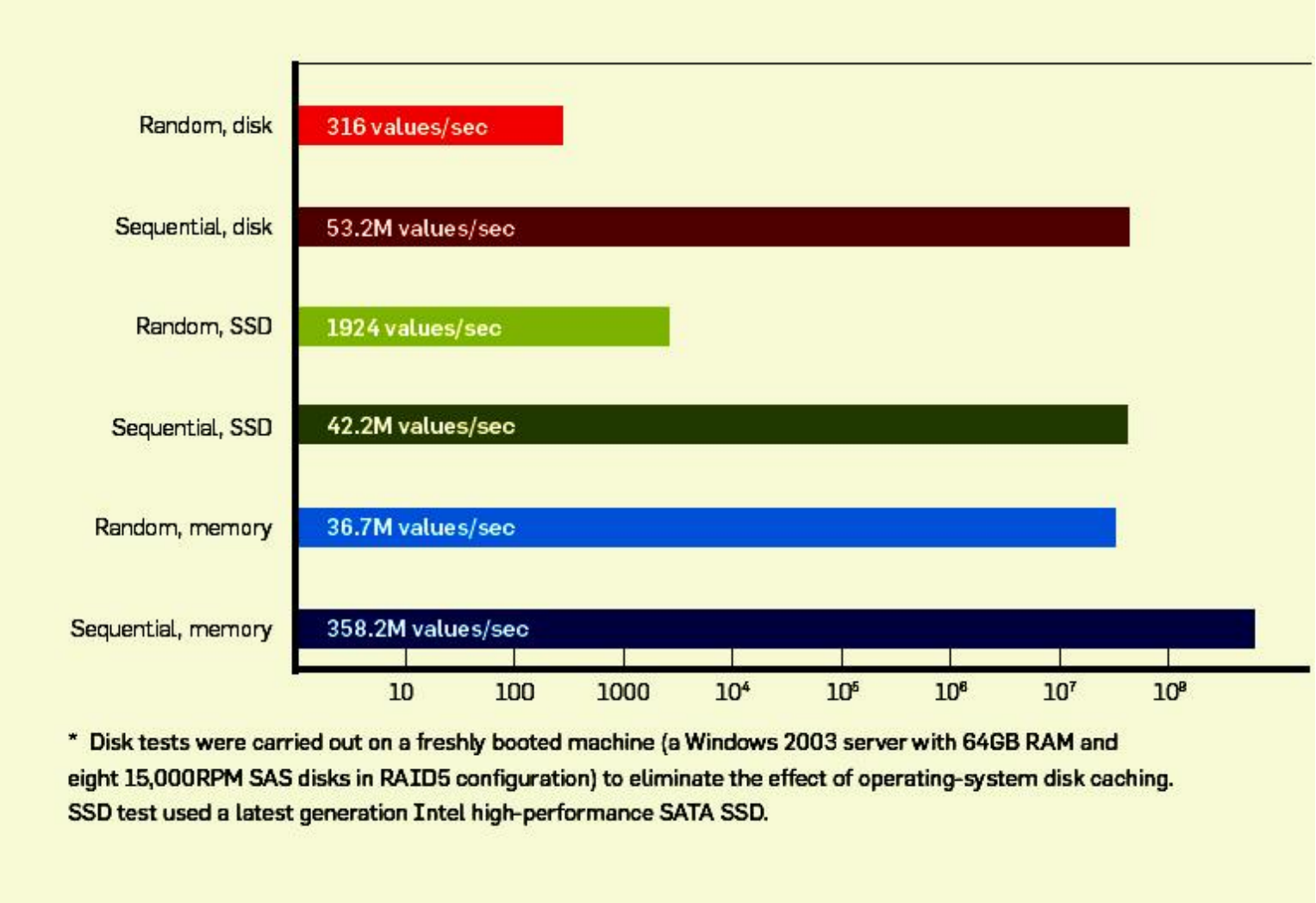
- *Disk I/O = read or write of a block* is very expensive compared with what is likely to be done with the block once it arrives in main memory.
  - Perhaps 1,000,000 machine instructions in the time to do one random disk I/O.

- The I/O model of computation measures the efficiency of an algorithm by counting how many disk reads and writes it needs.

- The unit of I/O is a **block** read/write

- The model is oversimplified – no difference between sequential and random access to several blocks

Assignment 1.1. Memory hierarchy



From: "Pathologies of Big Data", by Adam Jacobs, 2009