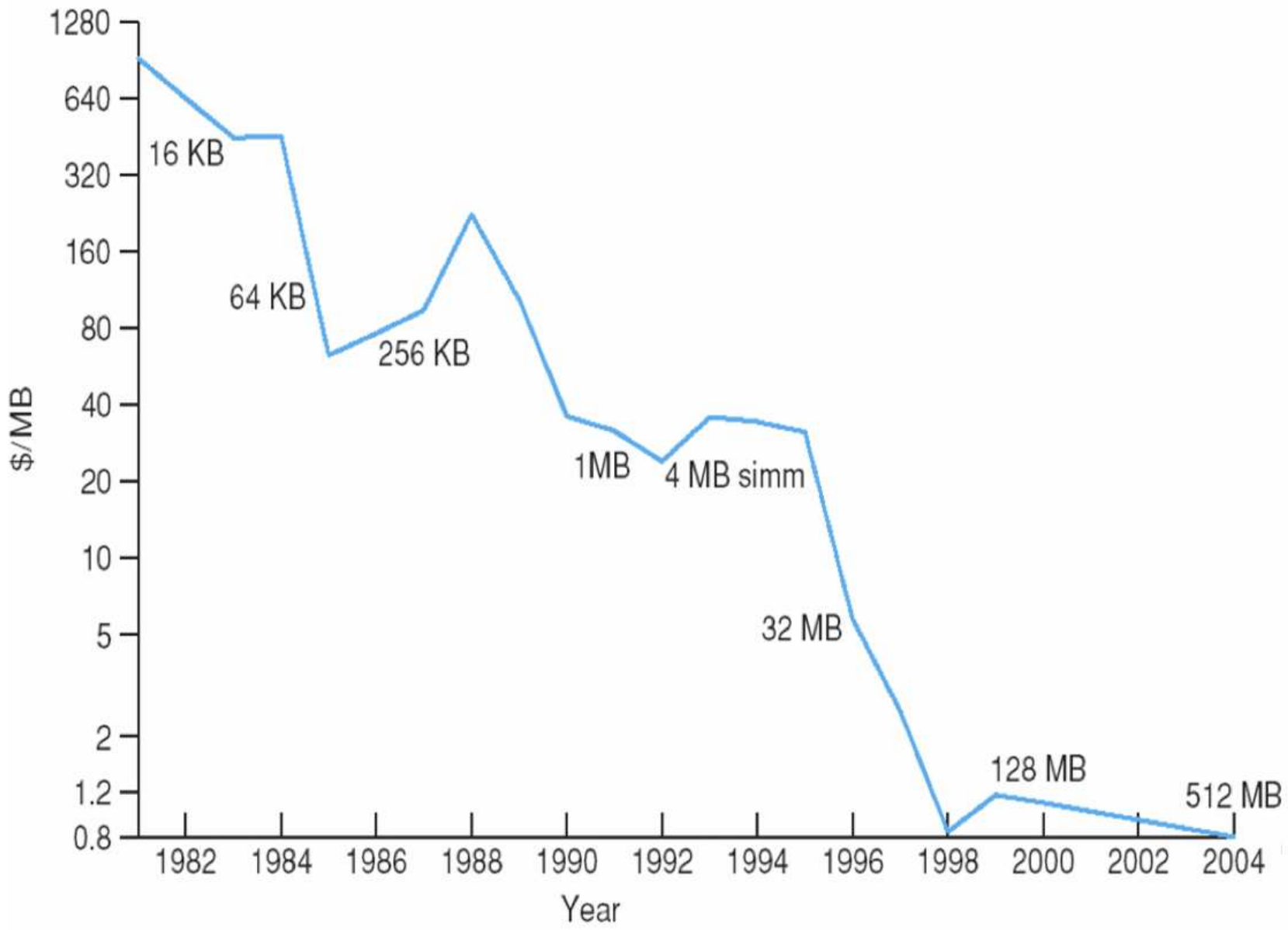
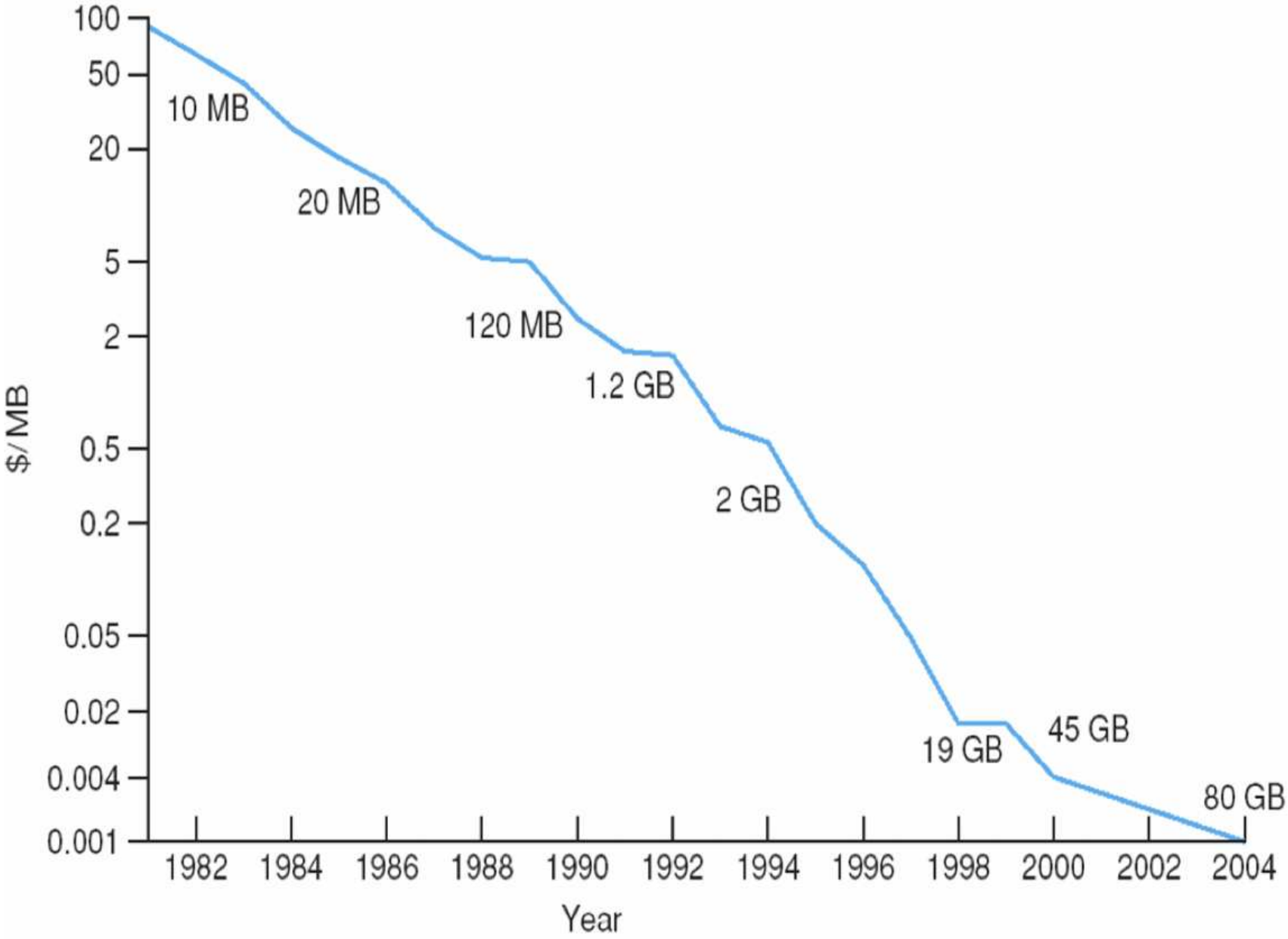
Memory

- Main memory — DRAM
- Secondary storage — disk, network
- Tertiary storage — tape

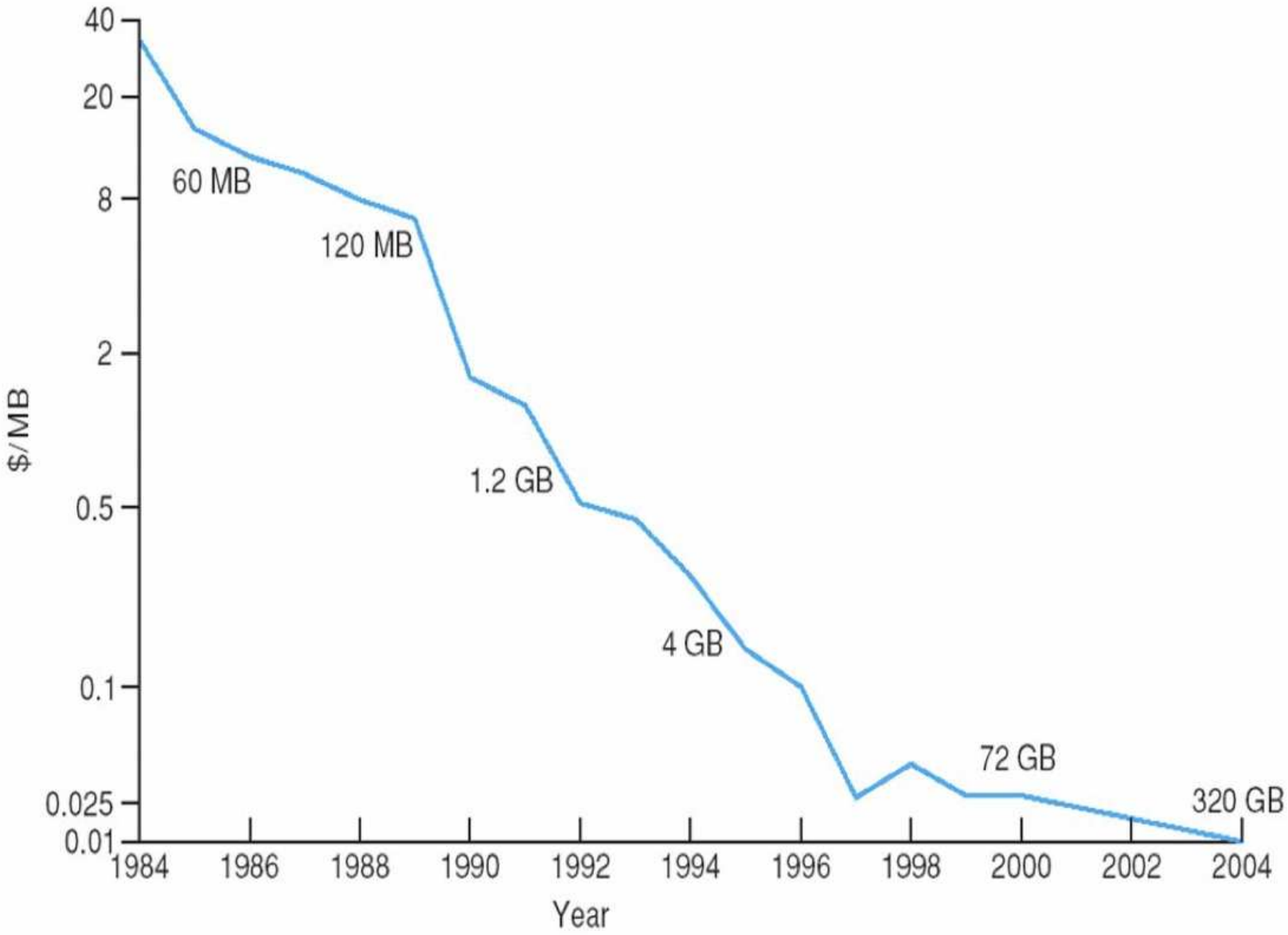
DRAM Prices


1280
640
320
160
80
40
20
10
5
2
1.2
0.8
$/MB
16 KB
64 KB
256 KB
1MB
4 MB simm
32 MB
128 MB
512 MB
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
Year

Disk Prices


100
50
20
5
2
0.5
0.2
0.05
0.02
0.004
0.001
$/MB
10 MB
20 MB
120 MB
1.2 GB
2 GB
19 GB
45 GB
80 GB
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
Year

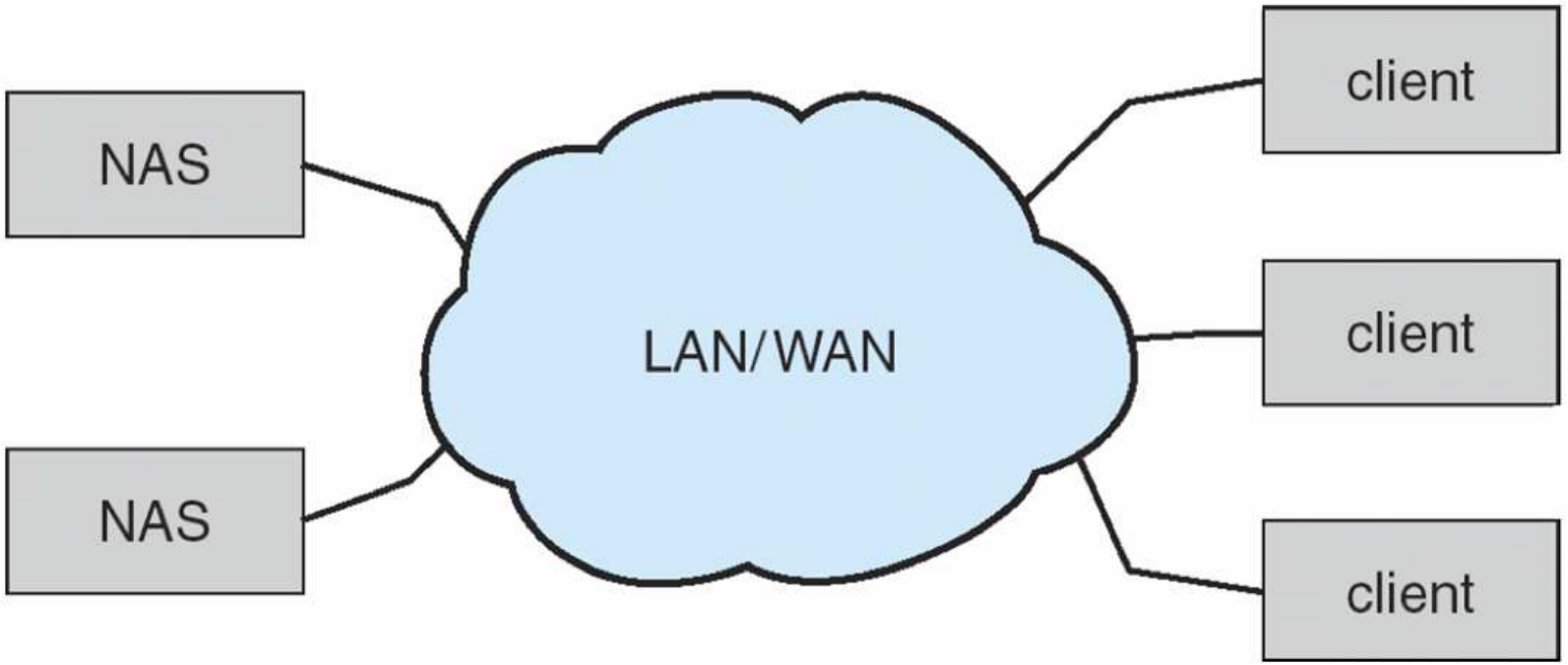
Tape Prices


40
20
8
2
0.5
0.1
0.025
0.01
$/MB
60 MB
120 MB
1.2 GB
4 GB
72 GB
320 GB
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
Year

Storage Choices

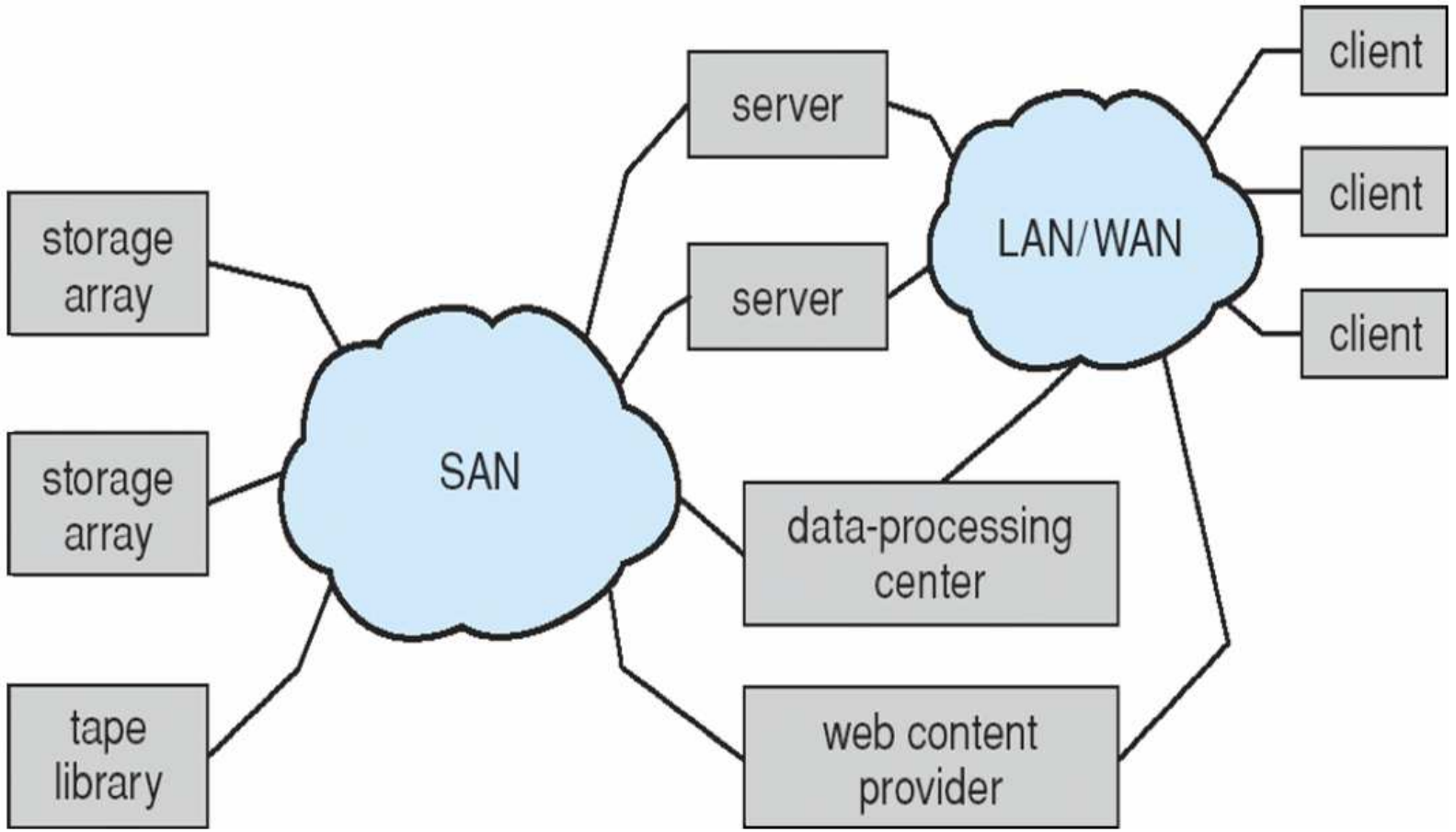
- Tape is now cheaper only if you need *lots* of tape
  - OSes designed around disk properties, instead
- Disk is still a lot cheaper than DRAM
  - NAND may change the equation

Network-Area Storage (NAS)

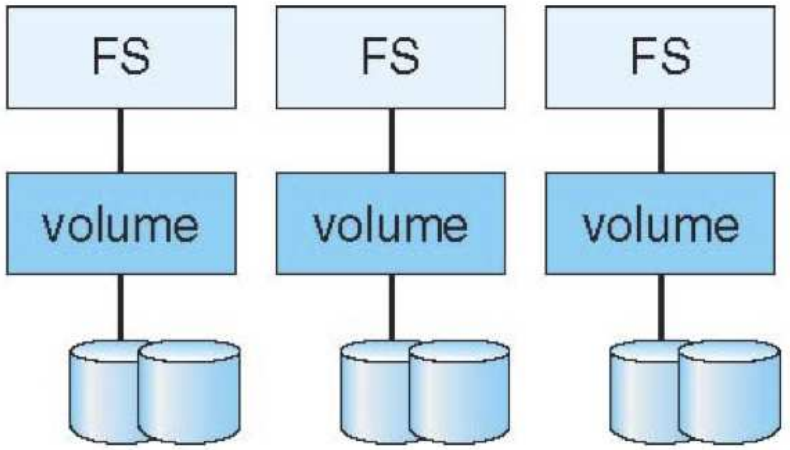


e.g., NFS

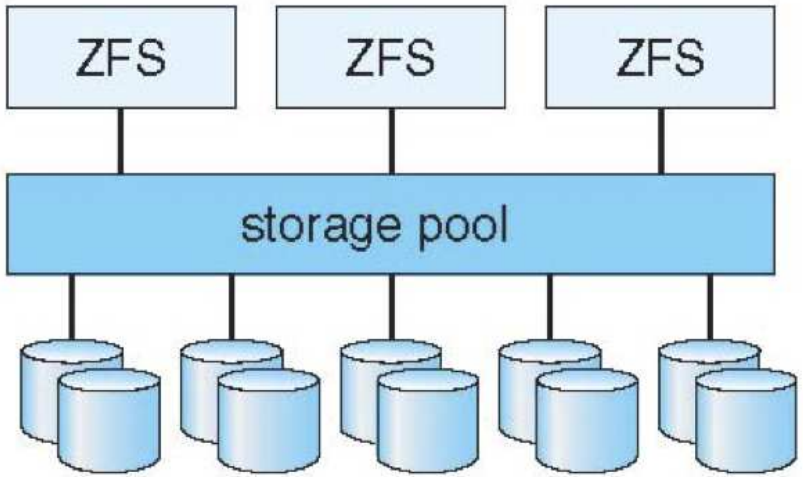
Storage-Area Network (SAN)



Backing NAS with Disks

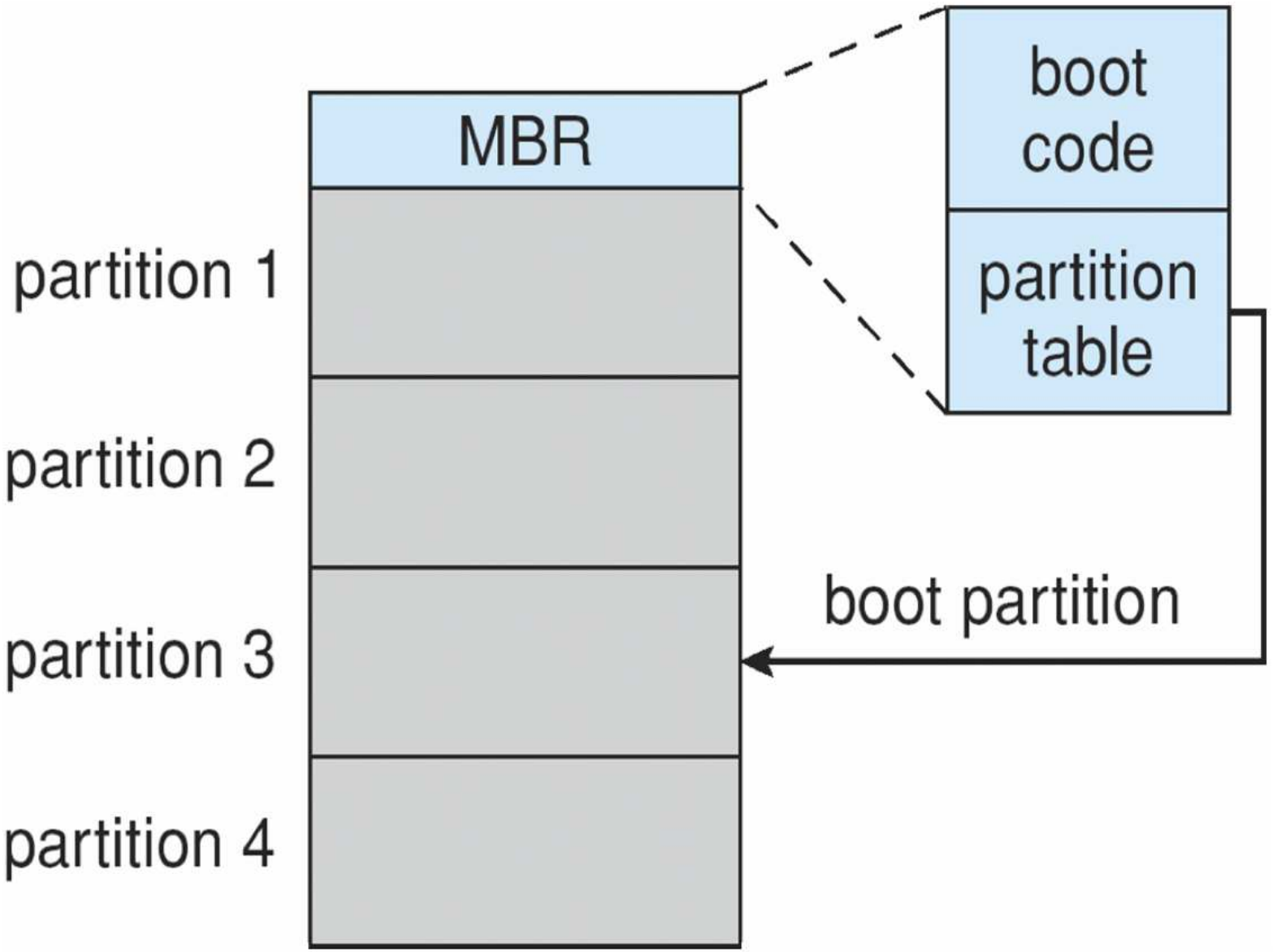


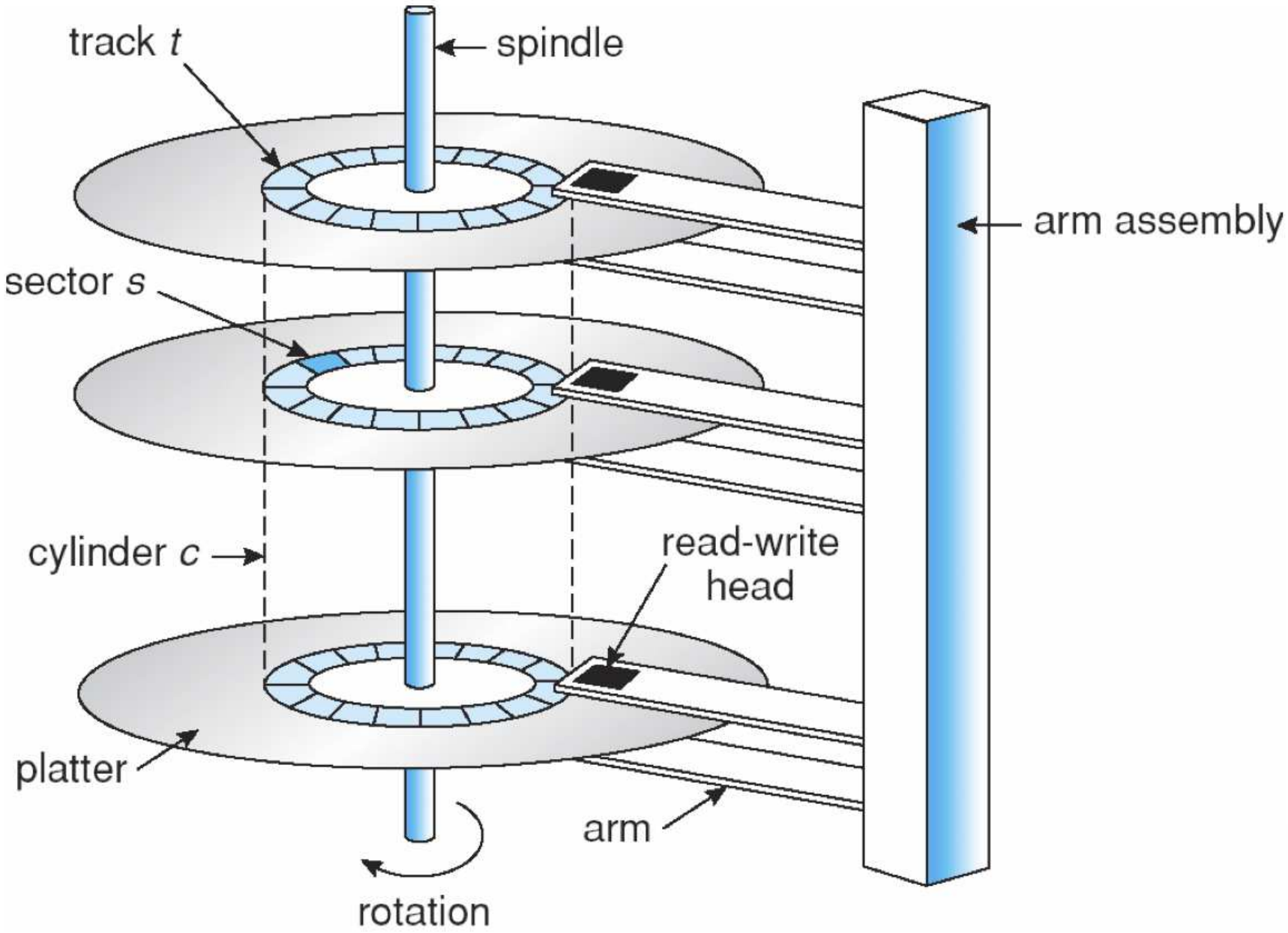
(a) Traditional volumes and file systems.

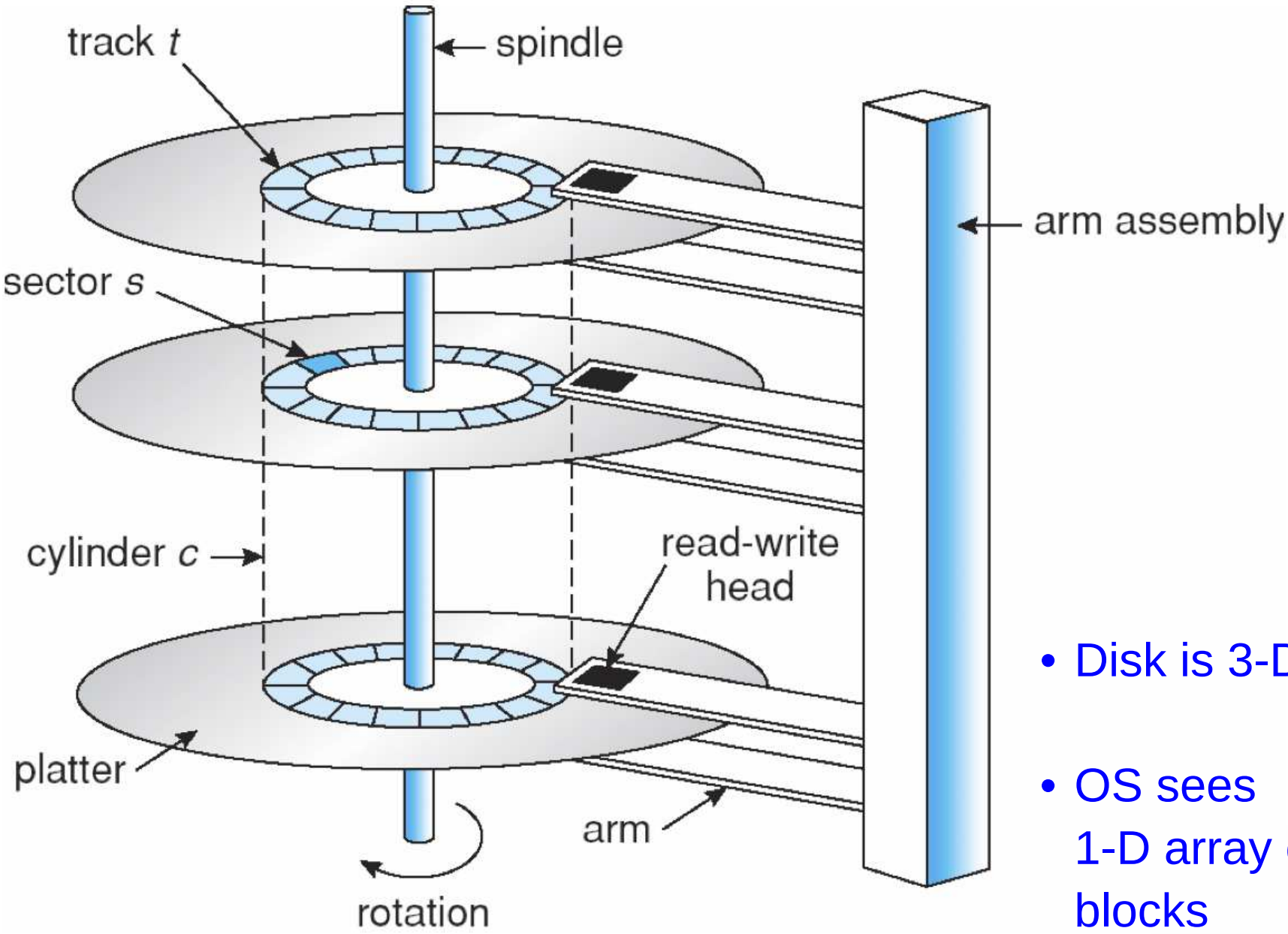


(b) ZFS and pooled storage.

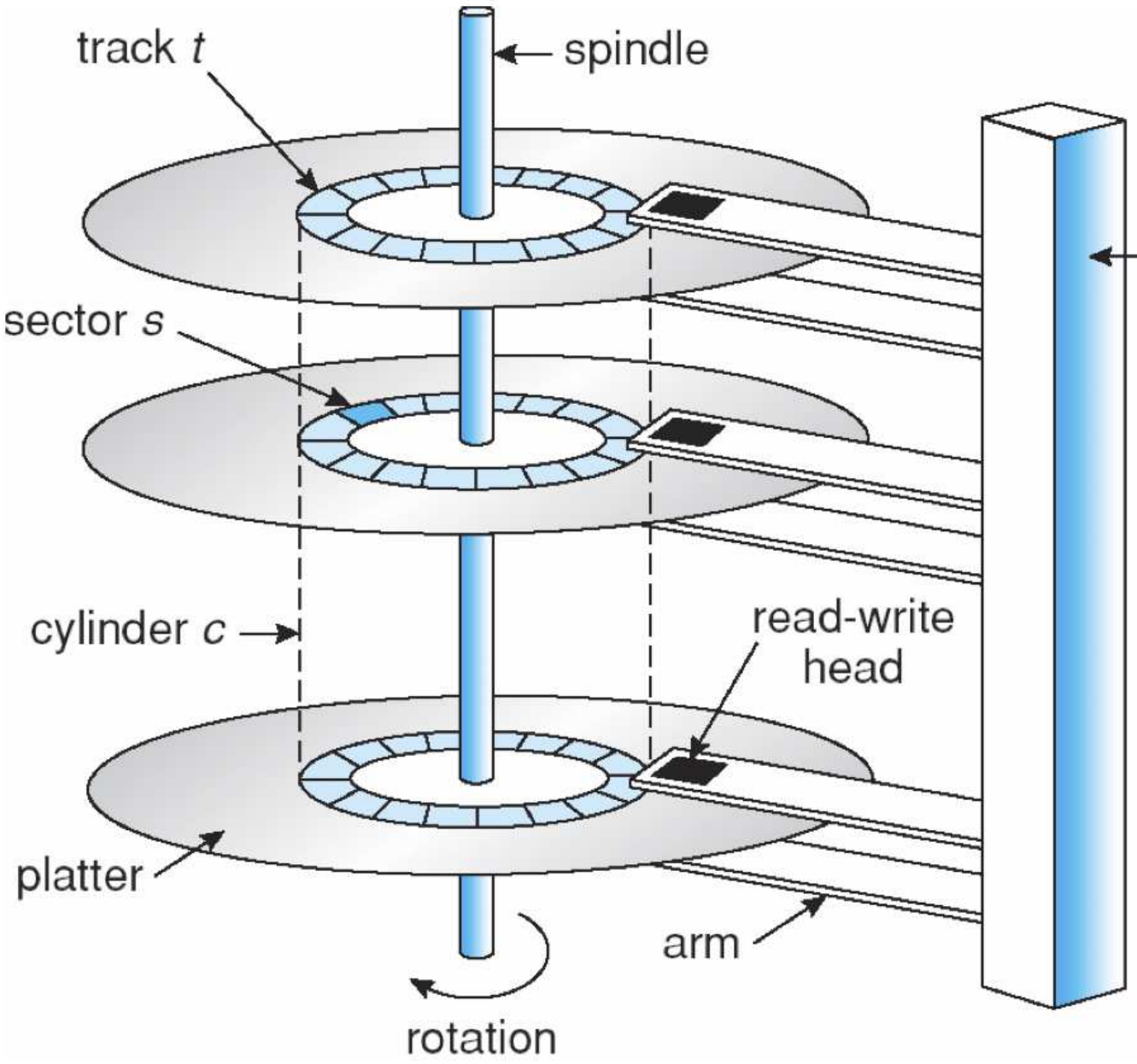
Local Disk Partitioning




track t
spindle
arm assembly
sector s
cylinder c
read-write
head
platter
arm
rotation



- Disk is 3-D
- OS sees 1-D array of blocks



- ***Seek time*** = time to move head to cylinder
- ***Rotational latency*** = time to get to the right sector

Disk Scheduling

Suppose that you need to read the following blocks:

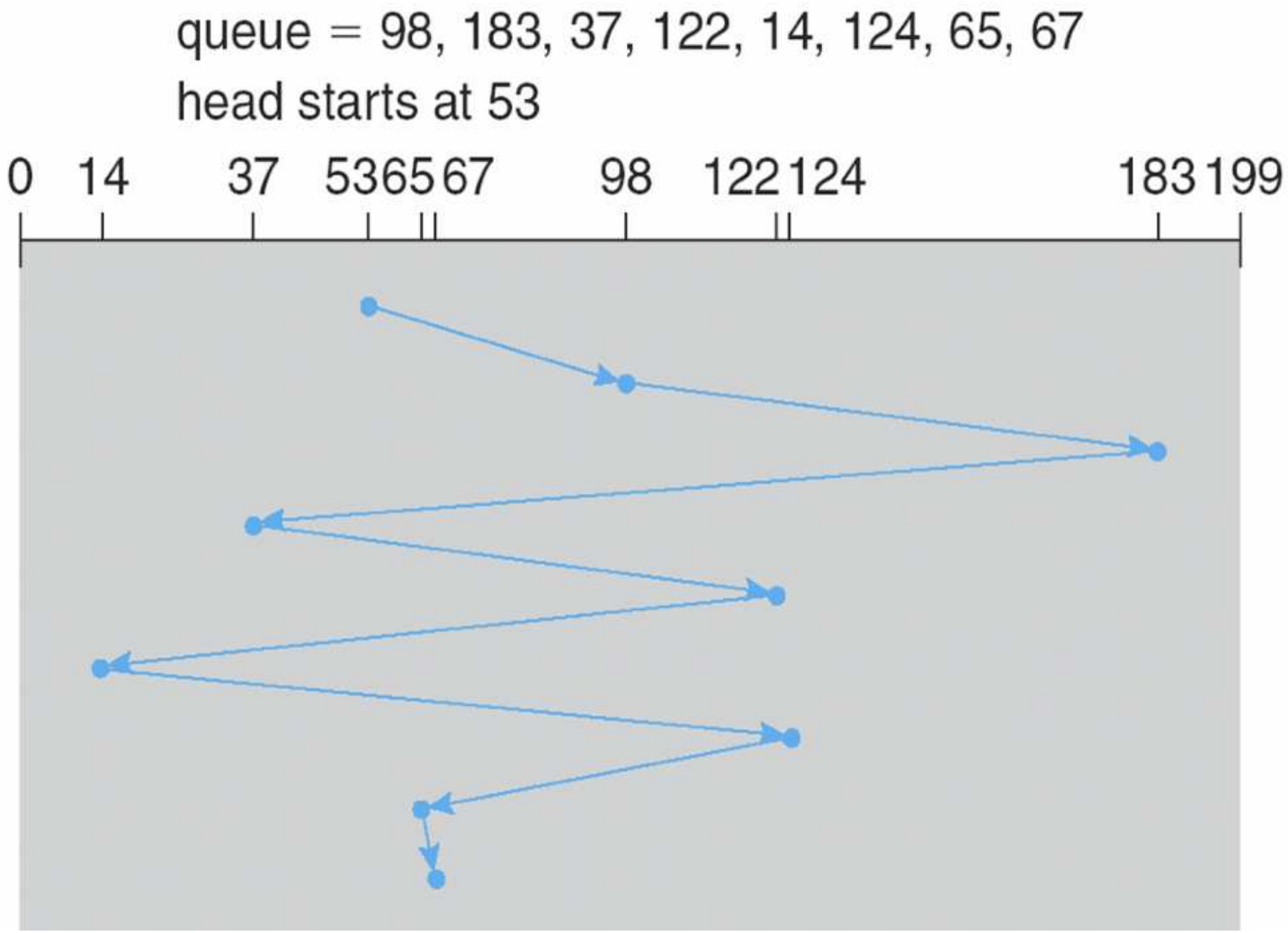
98, 183, 37, 122, 14, 124, 65, 67

and the dis heda is currently at block 53

Disk scheduling means picking an order to handle the current requests

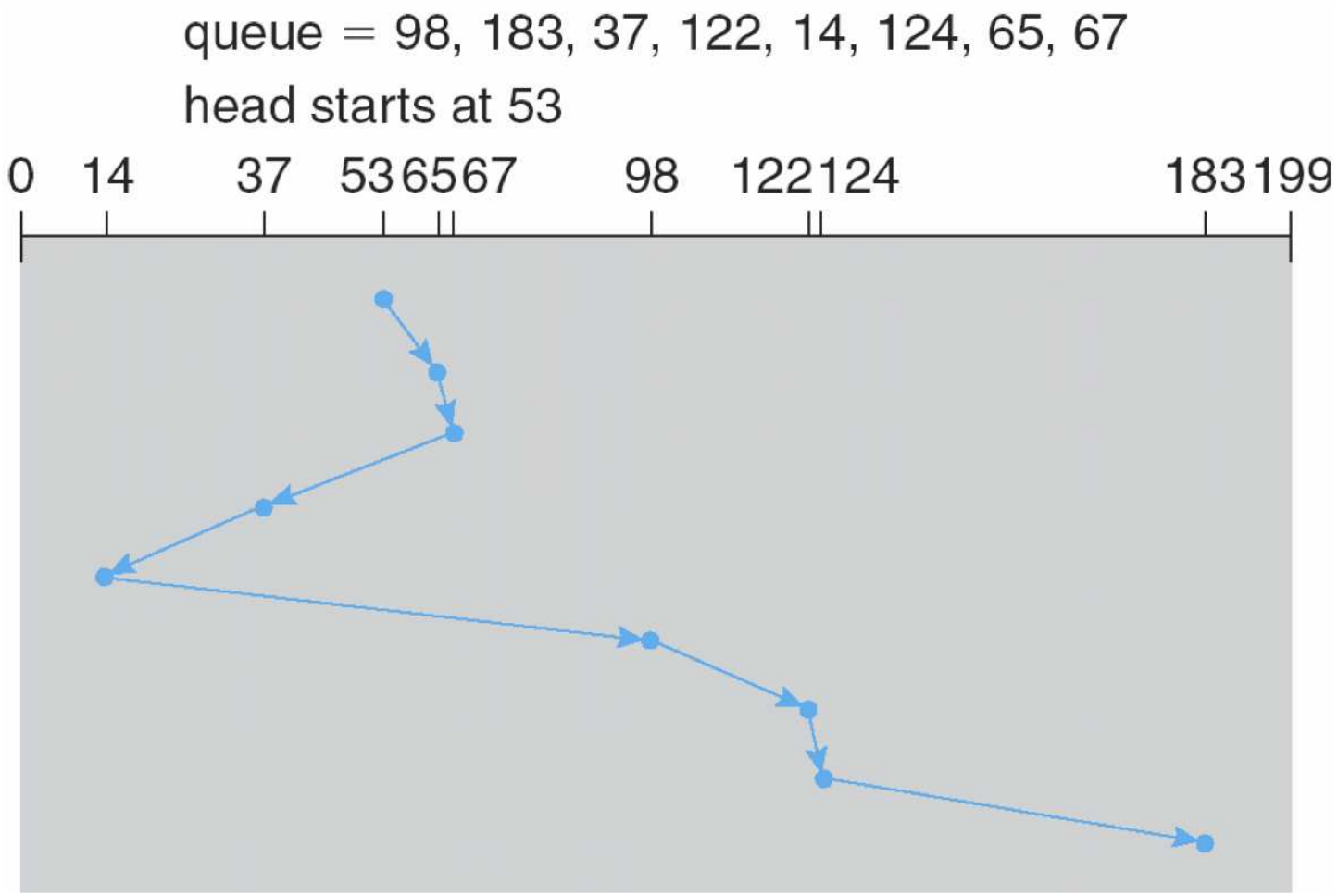
FCFS

First come, first served



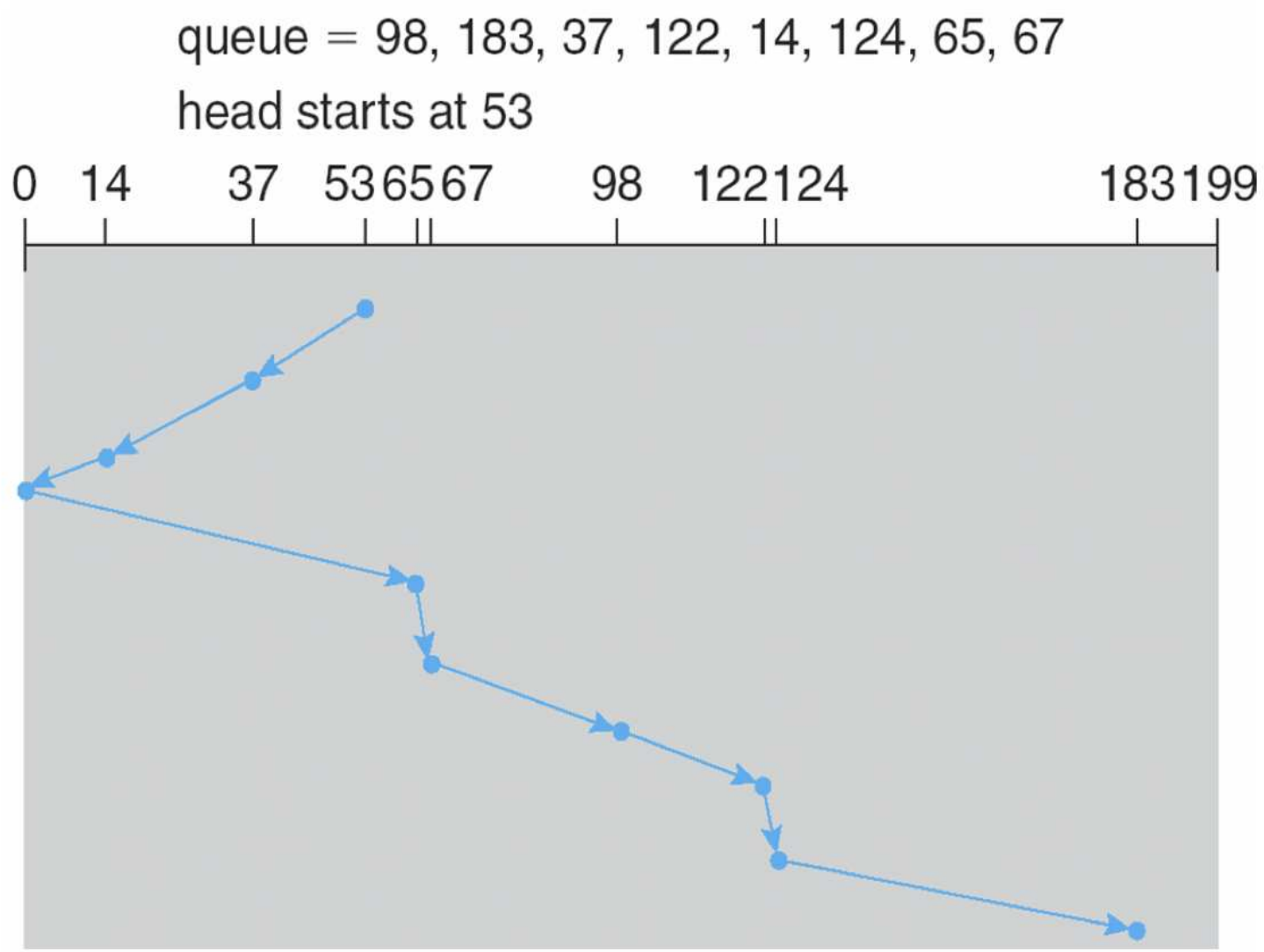
SSTF

Shortest seek time first



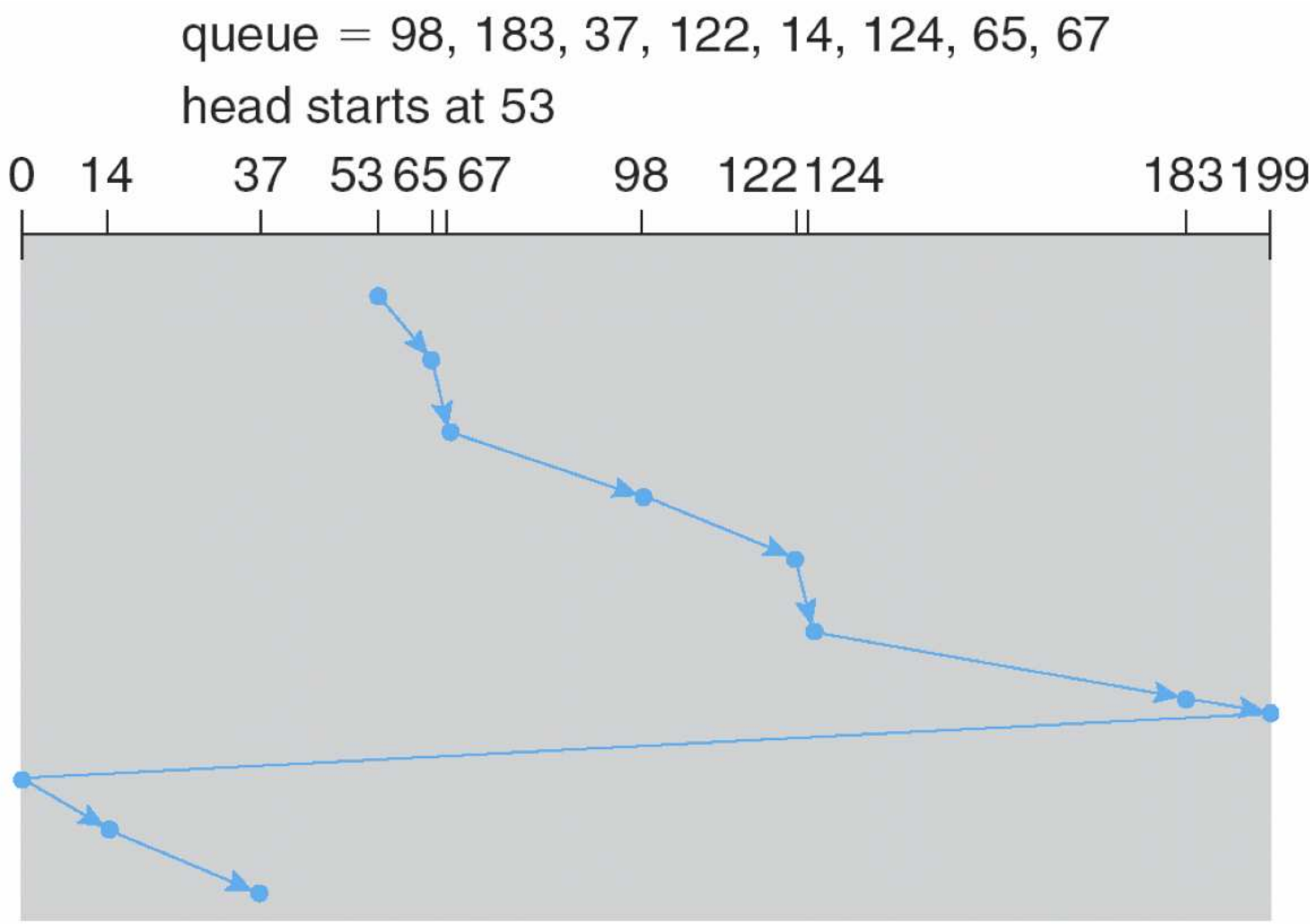
SCAN

Keep moving in one direction



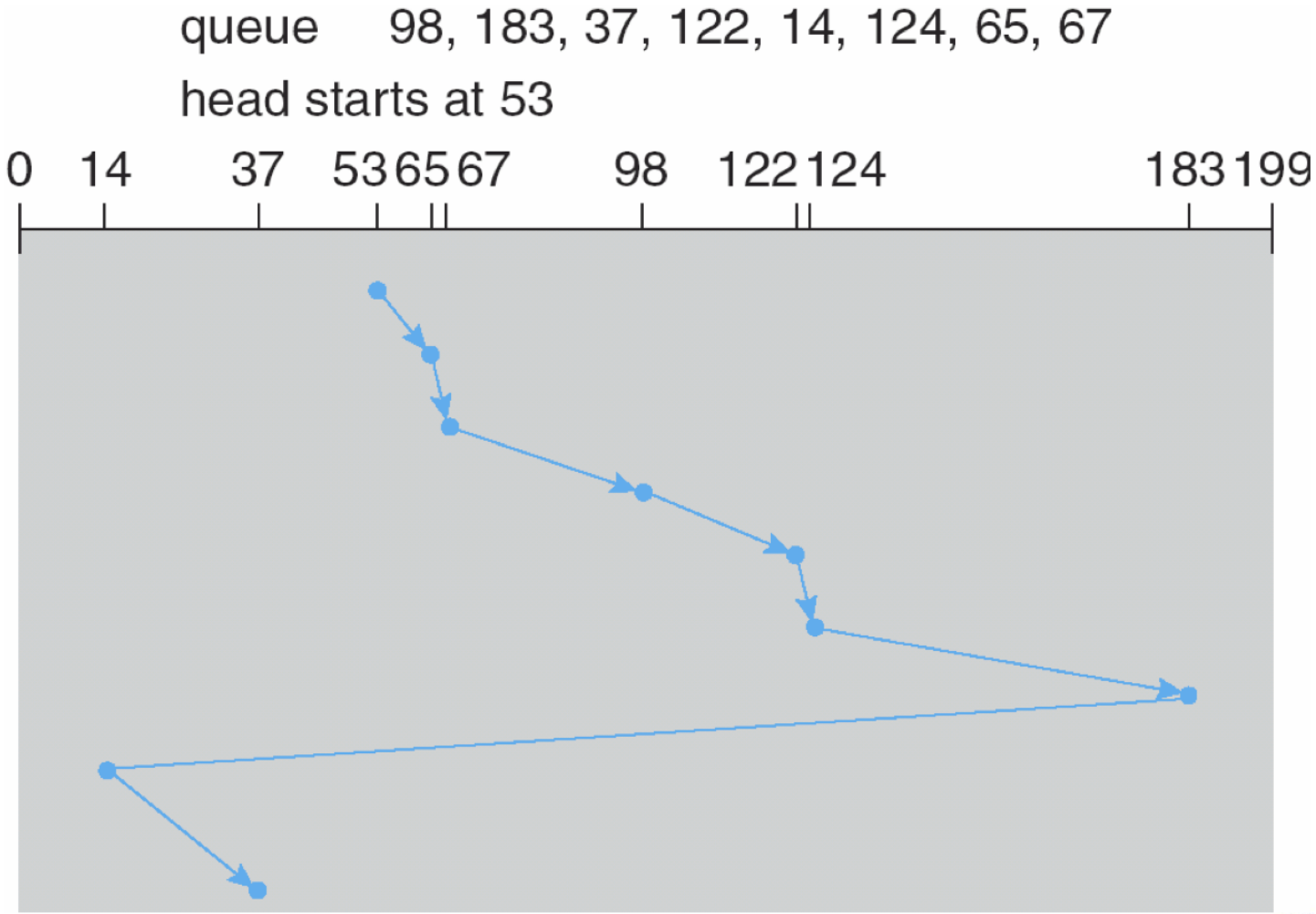
C-SCAN

Always move in one direction



C-LOOK

ONly go as far as requests need



Choosing an Algorithm

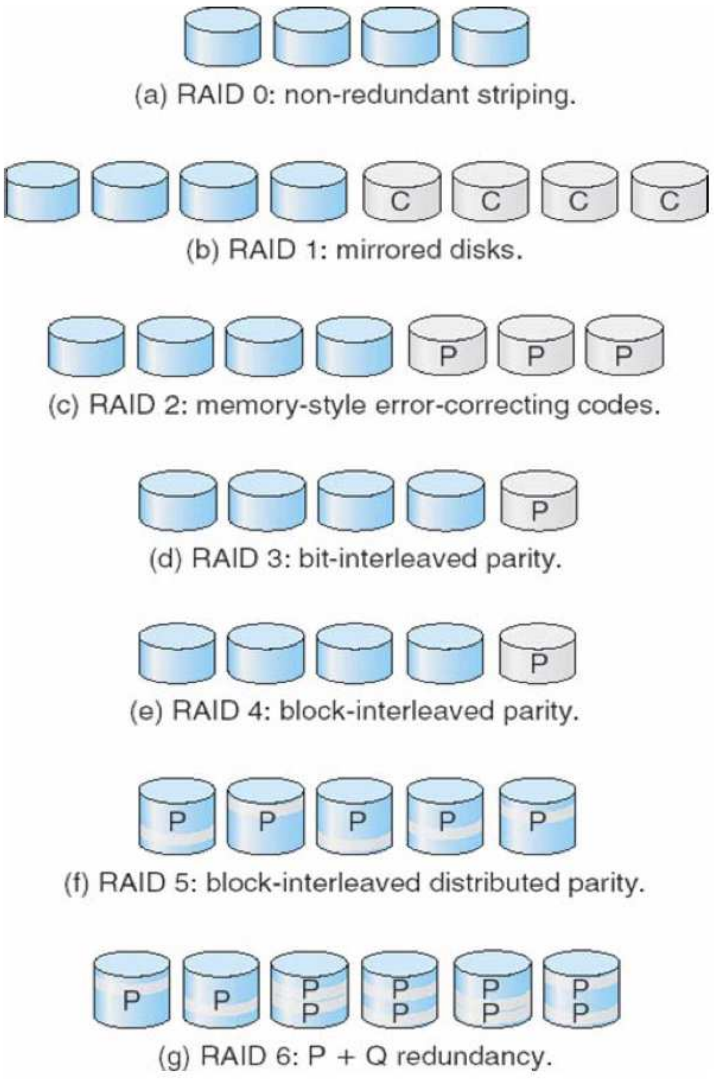
- SSTF is a popular choice
- SCAN and C-SCAN for systems where disk is used heavily

RAID

RAID = redundant array of independent disks

- Performance through parallelism
- Reliability through copies

RAID



(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c) RAID 2: memory-style error-correcting codes.

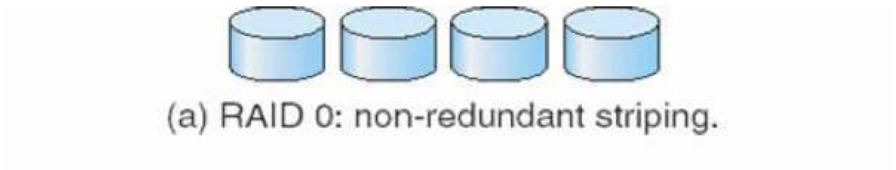
(d) RAID 3: bit-interleaved parity.

(e) RAID 4: block-interleaved parity.

(f) RAID 5: block-interleaved distributed parity.

(g) RAID 6: P + Q redundancy.

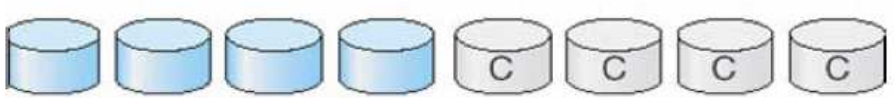
RAID 0



(a) RAID 0: non-redundant striping.

Multiple disks $\Rightarrow$ better preformance

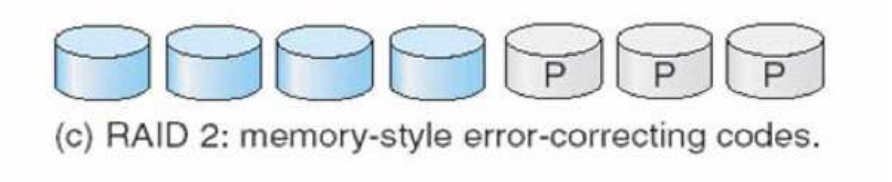
RAID 1



(b) RAID 1: mirrored disks.

Mirrored disks ⇒ better reliability

RAID 2

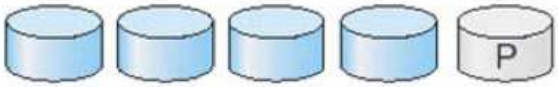


(c) RAID 2: memory-style error-correcting codes.

Use parity bits & ECC

- Better reliability
- Fewer disks than RAID 1

RAID 3

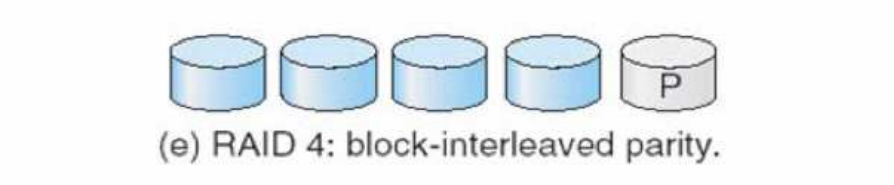


(d) RAID 3: bit-interleaved parity.

Single parity bit

- Assumes disk can detect its own bad blocks
- Fewer disks than RAID 2
- Slower writes

RAID 4



(e) RAID 4: block-interleaved parity.

Single parity bit

- Same space as RAID 3
- Easier to add disks

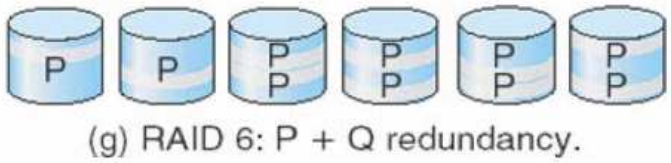
RAID 5

(f) RAID 5: block-interleaved distributed parity.

Distribute parity bits

- Same benefits as RAID 4
- Avoids bottleneck of a single parity-bit disk

RAID 6

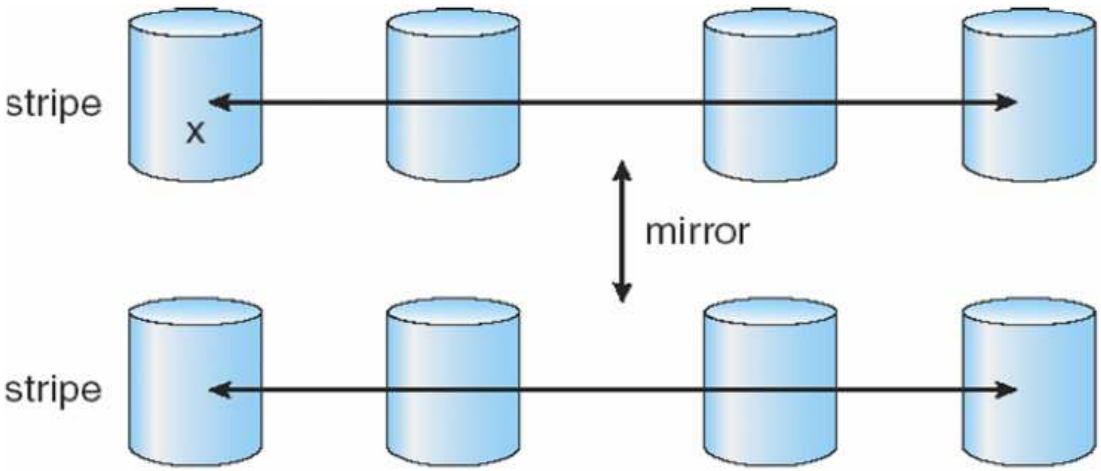


(g) RAID 6: P + Q redundancy.

Generalize parity bits to ECC

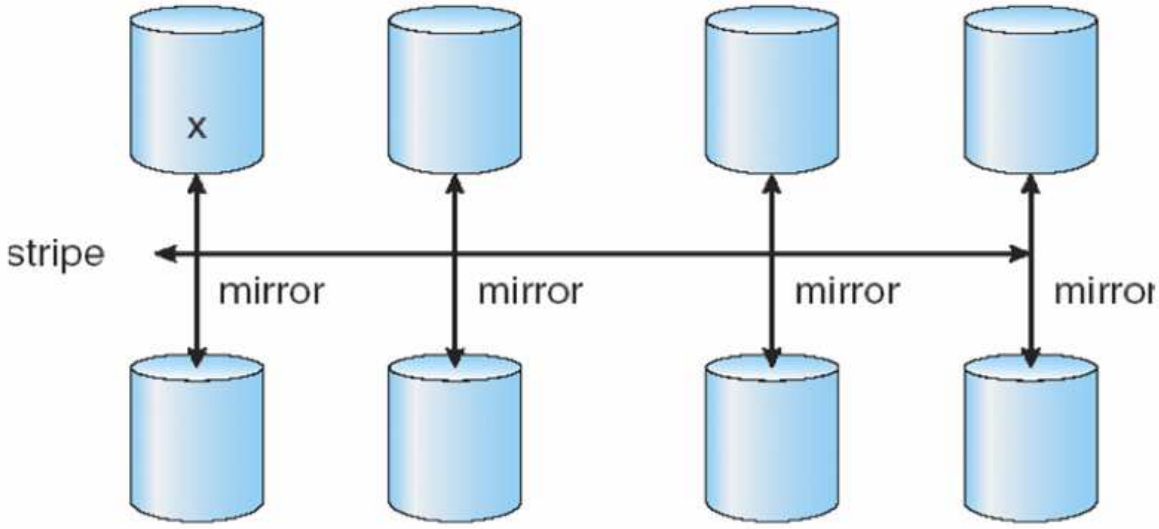
- Can survive multiple disk failures
- More storage than RAID 5

RAID 0+1



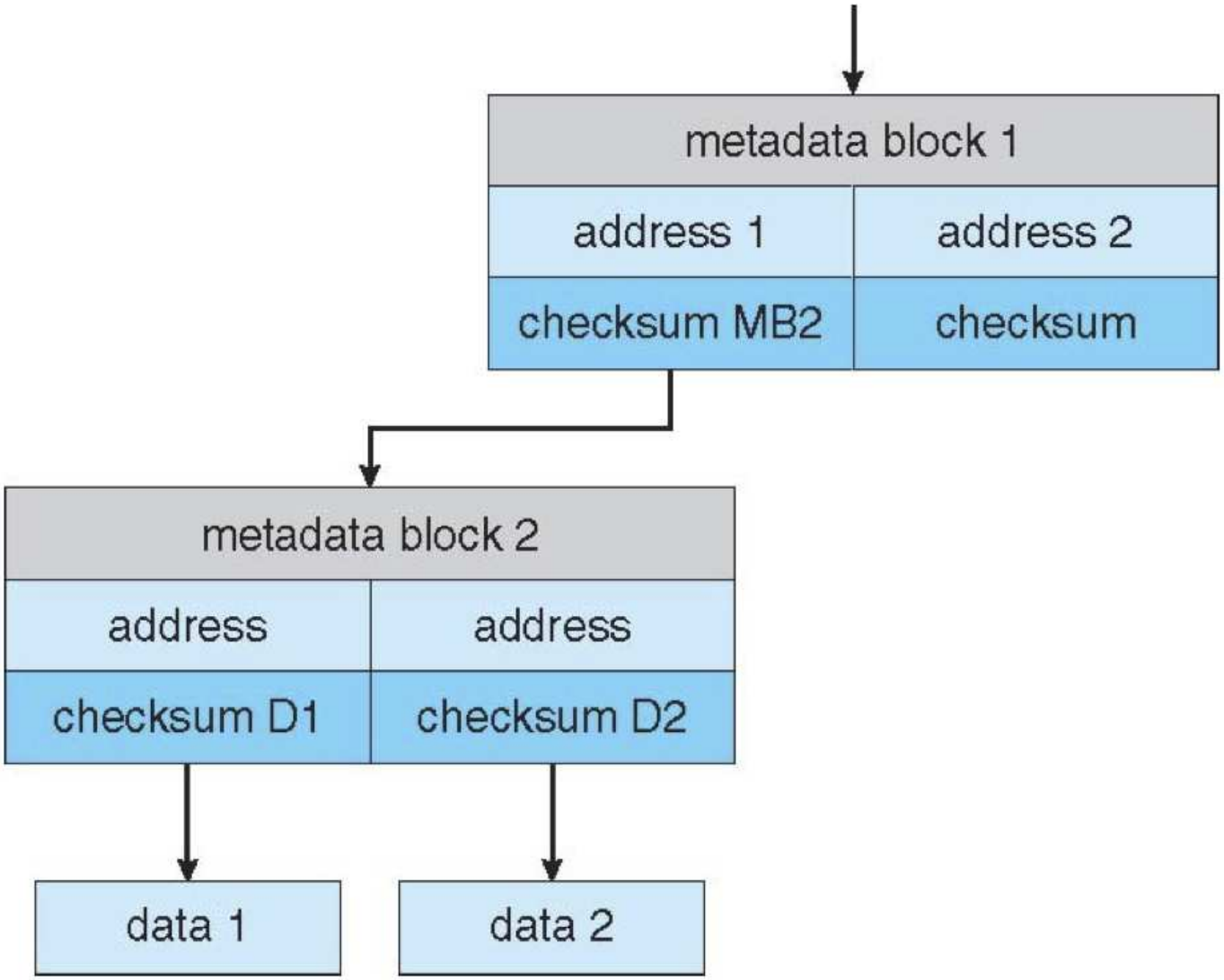
a) RAID 0 + 1 with a single disk failure.

RAID 1+0



b) RAID 1 + 0 with a single disk failure.

Checksums


metadata block 1
address 1
address 2
checksum MB2
checksum
metadata block 2
address
address
checksum D1
checksum D2
data 1
data 2

Summary

- Compared to main memory, disks are big and slow
- It’s worth complicating the OS to gain I/O performance
- Keep an eye on SSD developments, which make this all irrelevant