Operating Systems
CSE 511

Hard Drives
RAID

Instructor: Ruslan Nikolaev

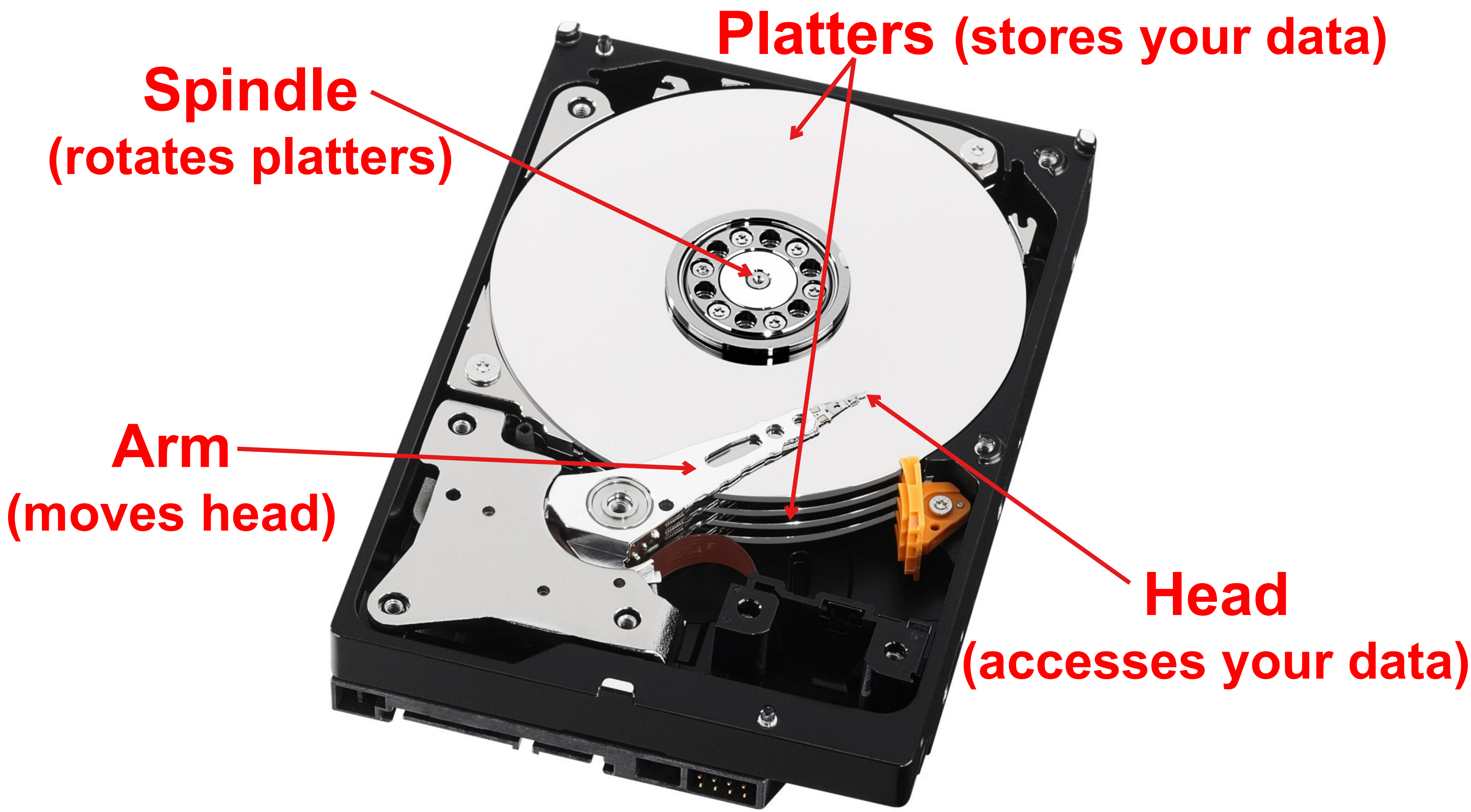
Grading Scale

- Our current grading scale is in our Syllabus
  - We use the standard PSU grading scale
  - Curve vs. do not curve
    - Not yet decided, mostly depends on the final exam grades
    - For assignments 1-2 only, it does not look to be necessary but some adjustments may be needed after assignment 3 and the final exam
  - Just do your best in assignment 3 and final exam!

SRTE Evaluation

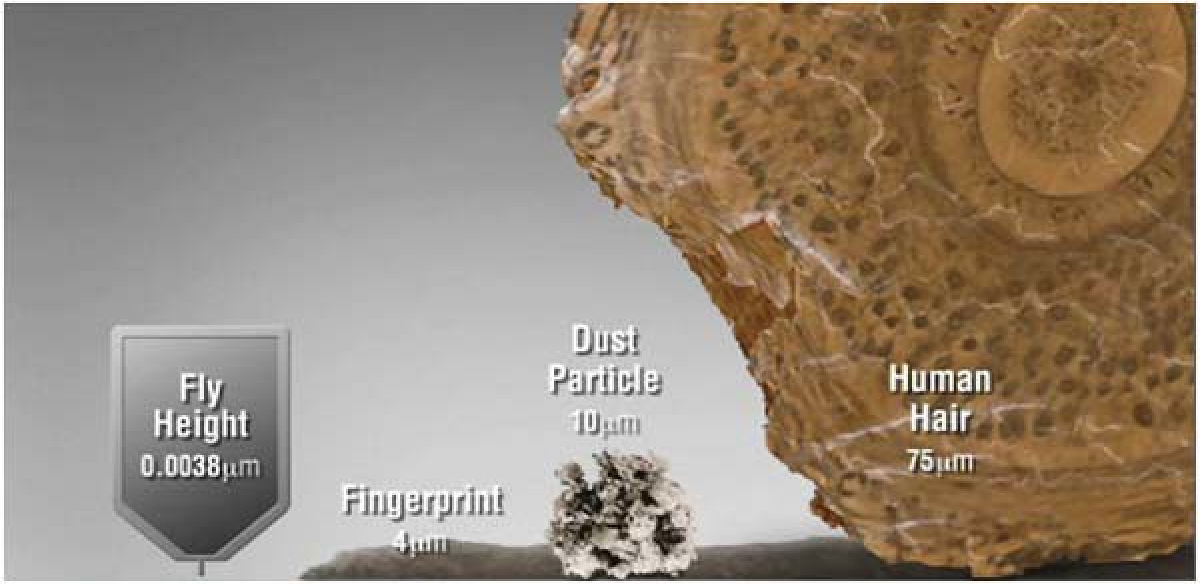
- Please submit the SRTE evaluation for CSE 511
  - You should have received the notification from the SRTE system
  - Your feedback is very important!
  - Should be available till Dec 11

Hard drive components

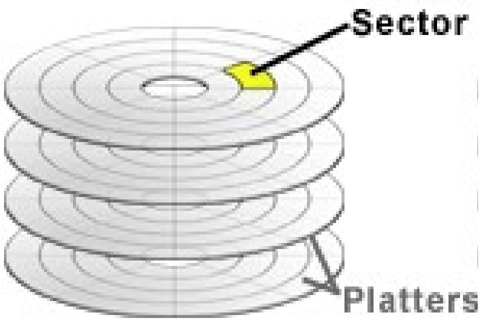


Hard drive operation

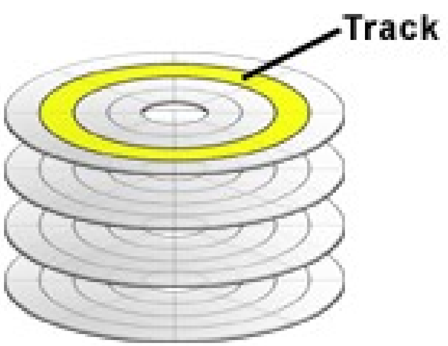
- Platters continuously rotate
  - Traditional: fixed speed; Power-savings: variable speed
- Arm moves head to inner/outer rings on platter
  - Very slow in computer time
- Head "hovers" above platter to read/write data



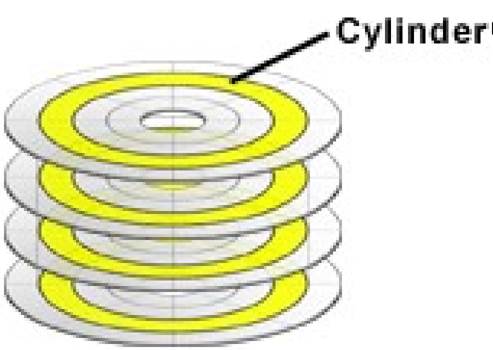
Hard drive geometry



- Sector – smallest unit (512 bytes)
  - Transitioning to 4kb (long process)

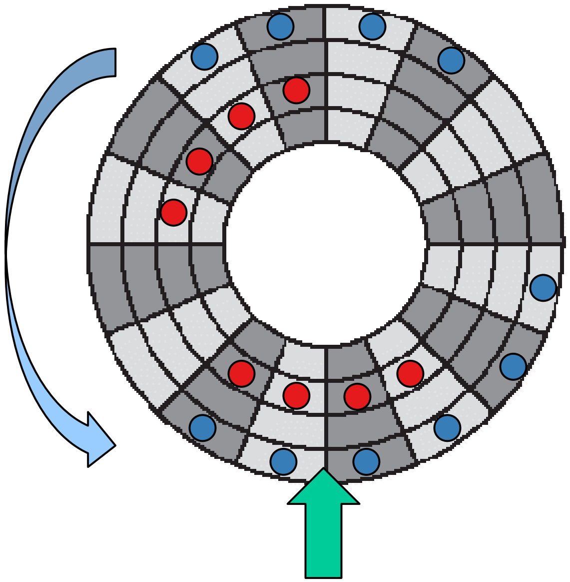


- Track – ring of sectors on a platter



- Cylinder – corresponding tracks across platters

Accessing data

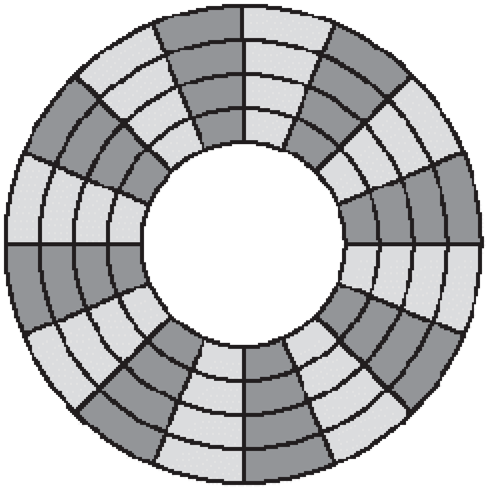


- Top-down view
  - Each region is a sector
  - Each ring of sectors is a track
- Data access latency
  - Seek time – time to switch track (depends on distance)
  - Rotation time – time to rotate platter (depends on rotation speed)
  - Transfer time – time to access data (depends on amount of data)

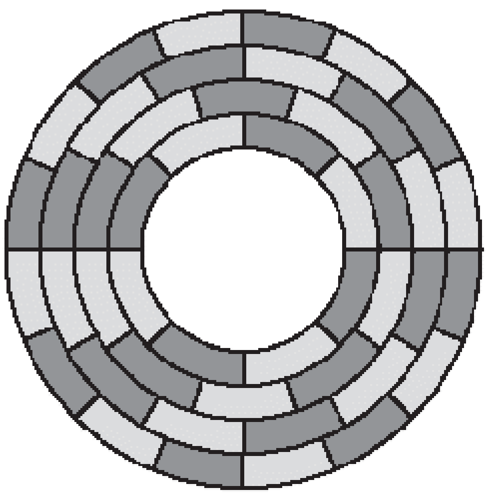
Is latency affected by read vs write access?
No

Is latency affected by rand vs seq access?
Yes

Density



- Top-down view
  - Each region is a sector
  - Each ring of sectors is a track
- What’s the problem with this image?
  - Outer track sectors are bigger



- Zoned-bit recording
  - Each sector roughly same size
  - More sectors on outer tracks
  - Faster reads/writes on outer tracks

Future of hard drives – cost

Hard drives are:

- Slow (vs SSDs)
- Noisy
- Power inefficient (vs SSDs)

<u>One reason to use hard drives</u>

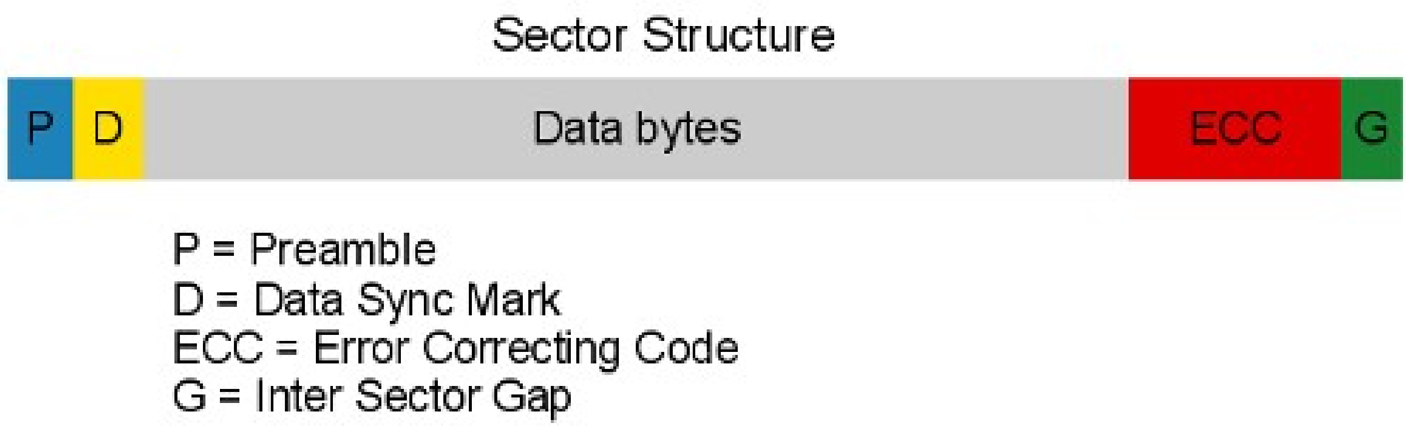
Cost

▭ Achieved via higher capacity & density

Handling errors (behind your back)

- Illusion: Hard drives perfectly store your data
- Reality: Hard drives make mistakes (due to density)

- Solution: Error-Correcting-Codes (ECC)

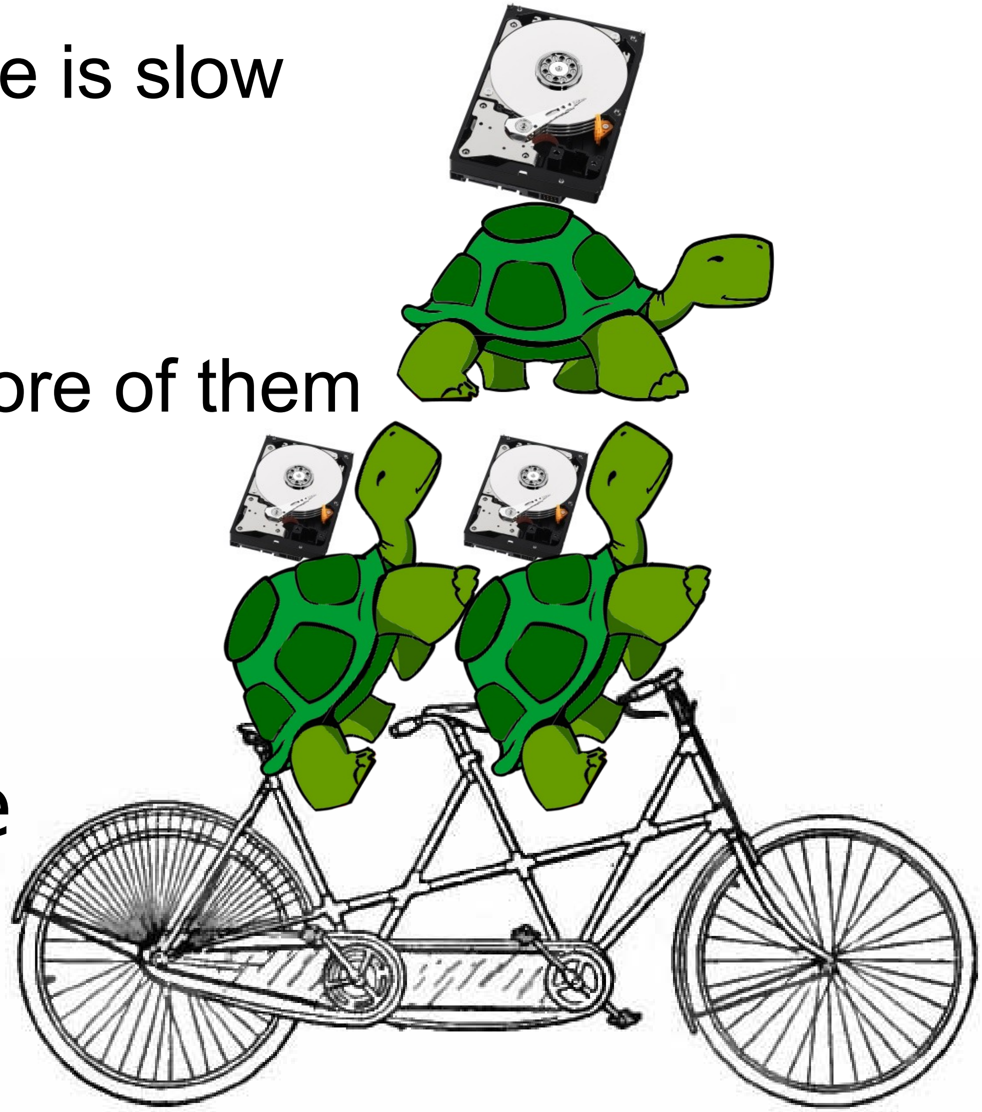


Error recovery

1. Use ECC to auto-correct errors
2. If it doesn't work, try again □ causes slowdowns
3. If all else fails, report sector as dead □ data lost
    - Hope you have redundancy (coming up next)
    - Now you have a dead sector: how to cope with hole?
    - Solution: remapping to spare sectors (every few tracks)

RAID - Storage is slow

- Problem: Storage is slow
- Solution: Use more of them



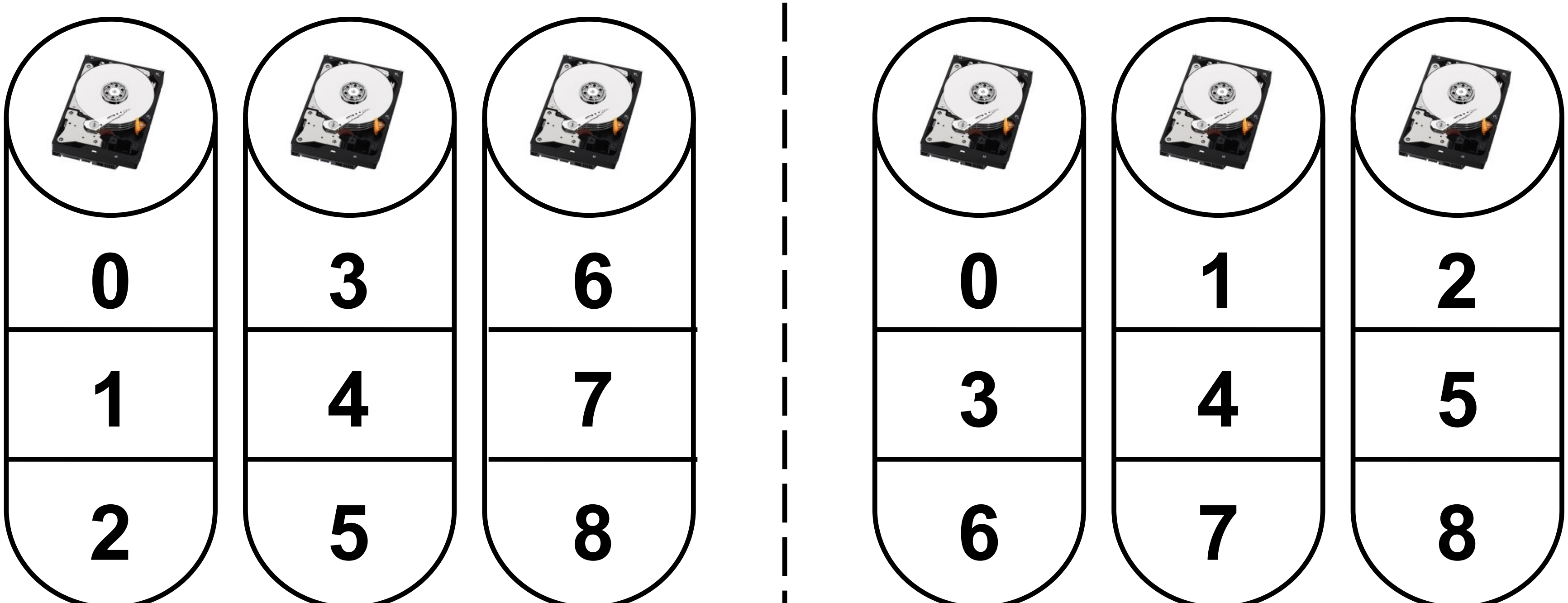
Redundant
Array of
Inexpensive
Disks

RAID0 – striping

- Striping – split your data across multiple disks

Option 1

Option 2



Failures

- MTBF – Mean Time Between Failures
- Example: 1.2 million hours **= 137 years!**

- Q: Am I done buying storage for life?
- A: No, marketing can fool the masses

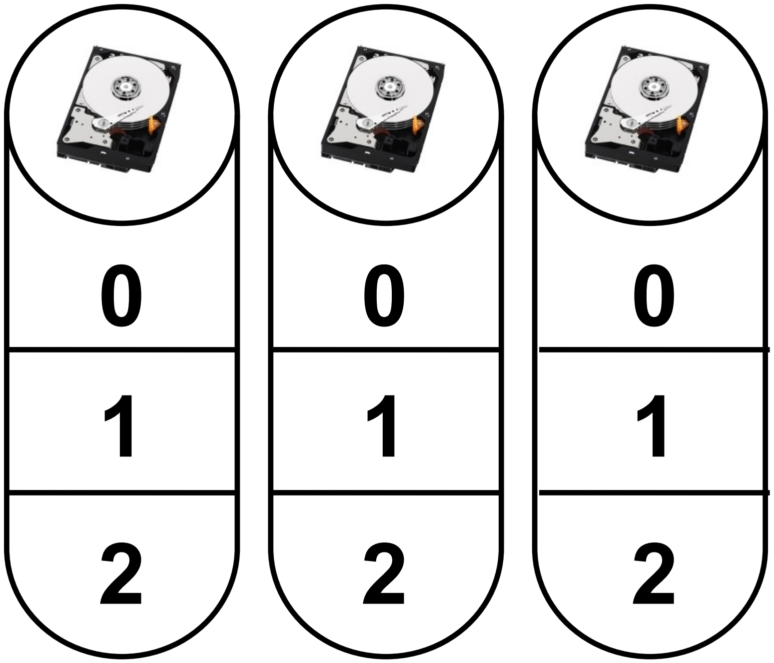
- Correct interpretation: failure rate = 1 / MTBF
- Q: What if you have a large 10,000 disk cluster?
- A: failure rate = 10,000 / MTBF = 1 / 5 days

⇒ **In expectation, a failure every 5 days!**

RAID1 – mirroring

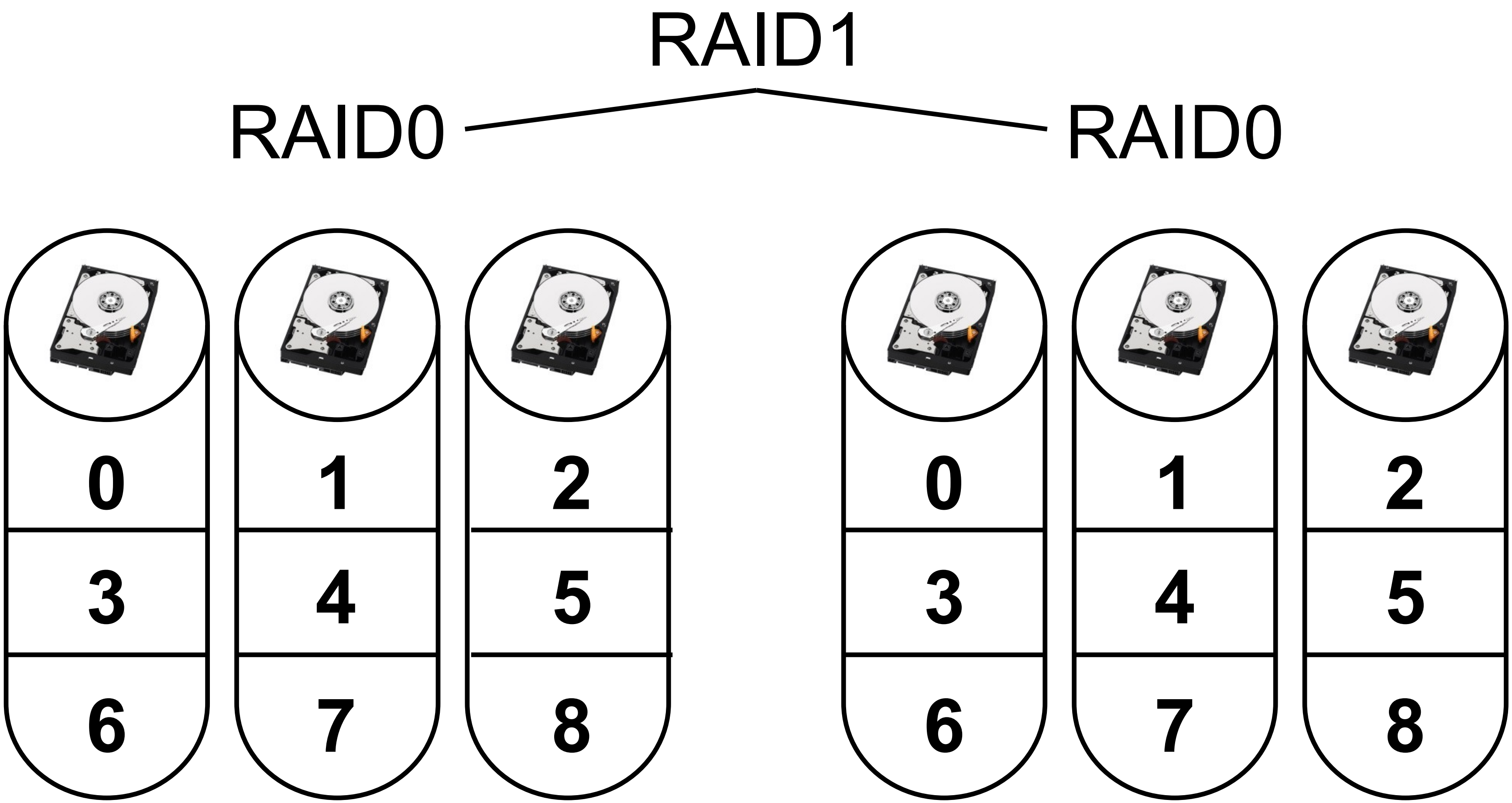
- Mirroring – keep redundant copies of data

- Pros: improved reliability
- Cons: expensive (uses a lot of space)



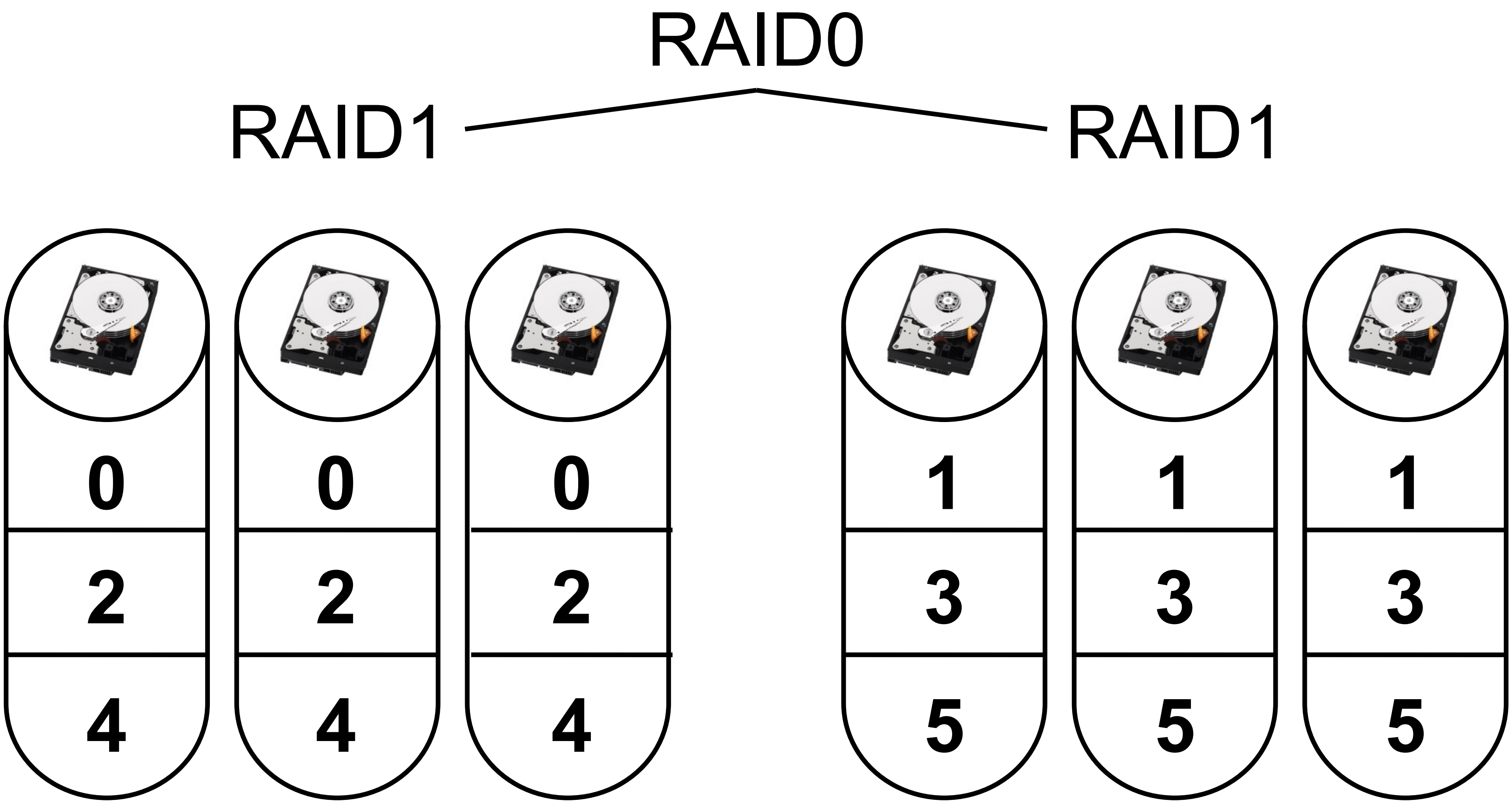
RAID0+1 – best of both worlds

- RAID01 = RAID0+1 = RAID0 + RAID1



RAID1+0 – best of both worlds v2

- RAID10 = RAID1+0 = RAID1 + RAID0



Counting the costs

- RAID1+0
- Pros:
  - Good performance
  - Good reliability
- Cons:
  - Expensive

- Q: Can we lower the cost?
- Solution: ECC – error correcting codes
  - ECC solution – parity

Parity

- Parity – determining even/odd number of 1's
  - Even = 0; Odd = 1
- How to calculate parity?

P=X^Y^Z

X	Y	Z	Even/Odd
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Recovering

- How to recover from failure?

X=Y^Z^P Y=X^Z^P Z=X^Y^P P=X^Y^Z

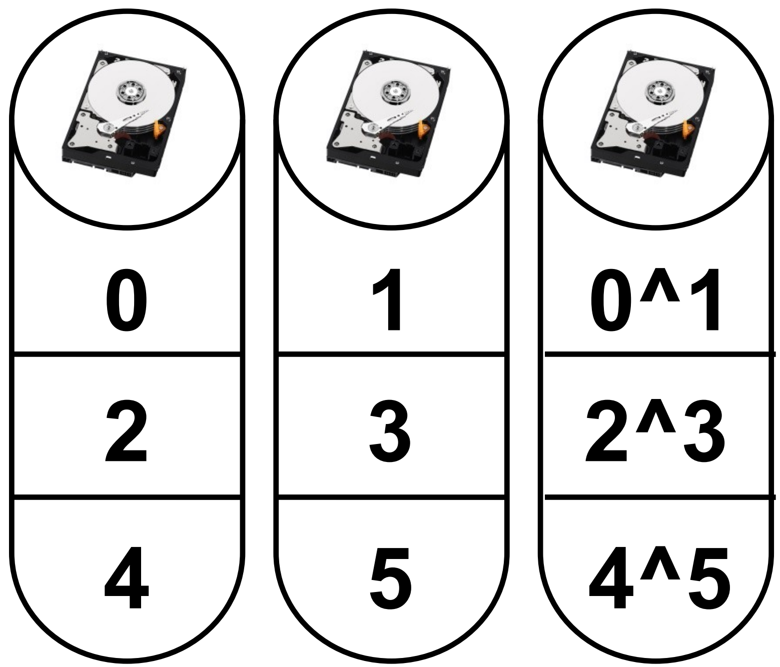
X	Y	Z	Even/Odd
0	~~0~~	0	0
0	~~0~~	1	1
0	~~1~~	0	1
0	~~1~~	1	0
1	~~0~~	0	1
1	~~0~~	1	0
1	~~1~~	0	0
1	~~1~~	1	1

Recovering

- How to recover from failure?
- Proof (of Y = X^Z^P):
- X^Z^P
- = X^Z^(X^Y^Z) — Replace P by def of parity
- = (X^X)^Y^(Z^Z) — xor is associative and commutative
- = 0^Y^0 — Anything xor itself is 0
- = Y — 0 xor anything is that value

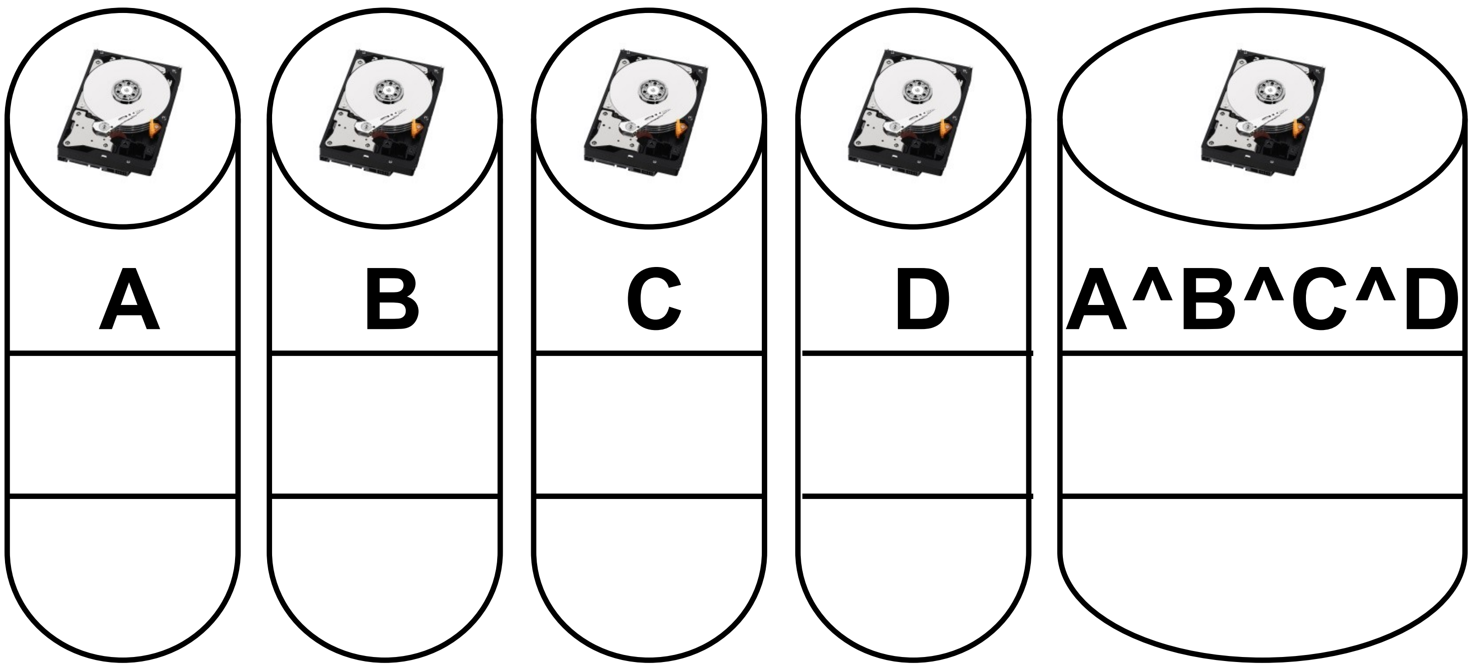
RAID4 – parity disk

- Keep a disk of parity rather than replicate all data
- Pros: good reliability at lower cost
- Cons: not as good performance



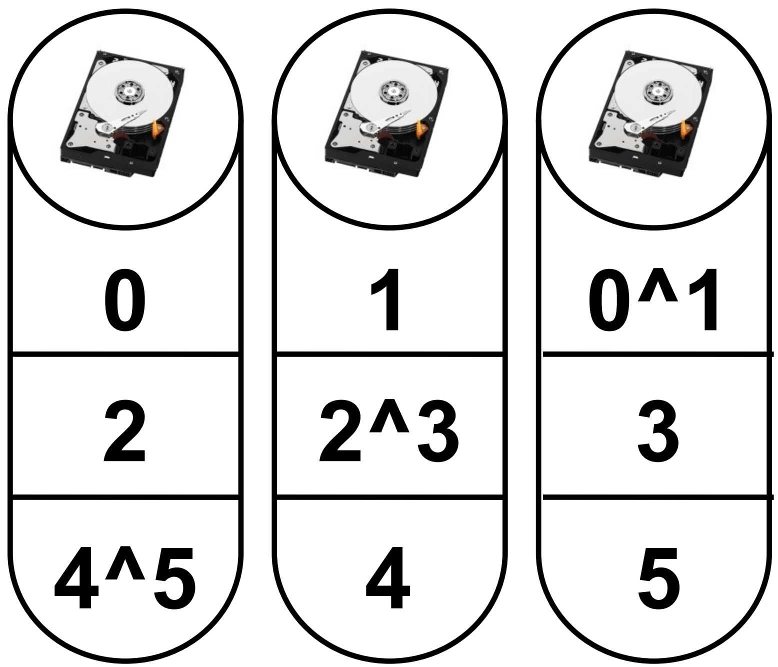
Writing new data

- Suppose we want to change A → A'
- Steps: Always access parity → bottleneck
  - ~~Read B, Read C, Read D~~ Read A, Read A^B^C^D
  - ~~Write A', Write A'^B^C^D~~ Write A', Write A'^A^(A^B^C^D)



RAID5 – striped parity

- Stripe the parity blocks to balance load
- Pros: good reliability, low cost, good performance
- Cons: can only tolerate a single failure



<u>RAID6</u>

Tolerates up to 2 failures using advanced ECC (Reed-Solomon)