# CS 423
# Operating System Design: Persistence: RAID
# 04/11

Ram Alagappan

# AGENDA / LEARNING OUTCOMES

Why do we need more than one disk in a system?

How do we achieve resilience against disk errors?

# RECAP

# SEEK, ROTATE, TRANSFER

Seek cost:

- Not purely linear cost
- Must accelerate, coast, decelerate, settle
- Settling alone can take 0.5 - 2 ms

Entire seeks often takes 4 - 10 ms

Average seek = 1/3 of max seek

Depends on rotations per minute (RPM)

- 7200 RPM is common, 15000 RPM is high end

Average rotation – half of a rotation

HDDs

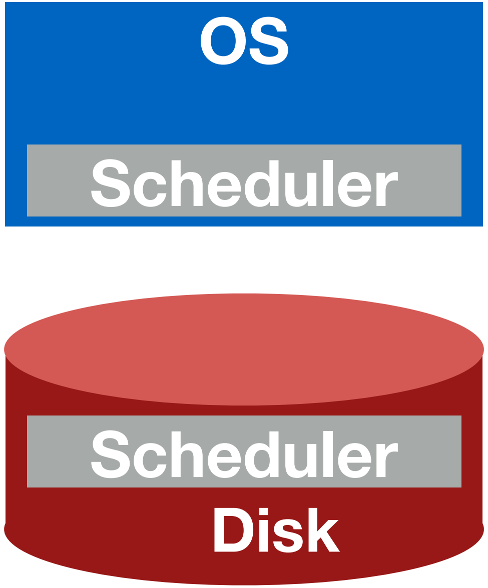
100+ MB/s is typical for maximum transfer rate

# I/O SCHEDULERS

Given a stream of I/O requests, in what order should they be served?

Position of disk head relative to request position matters more than length of job

# SCHEDULERS




Where should the scheduler go?

# WORK CONSERVATION

Work conserving schedulers always try to do work if there's work to be done

Sometimes, it's better to wait instead if system anticipates another request will arrive

Possible improvements from I/O Merging

# END RECAP

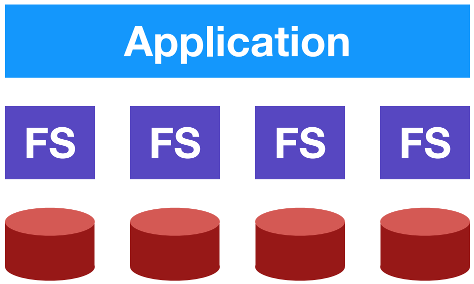
# ONLY ONE DISK?

Sometimes we want many disks — why?

- capacity
- reliability
- performance

Challenge: most file systems work on only one disk

# SOLUTION 1: JBOD

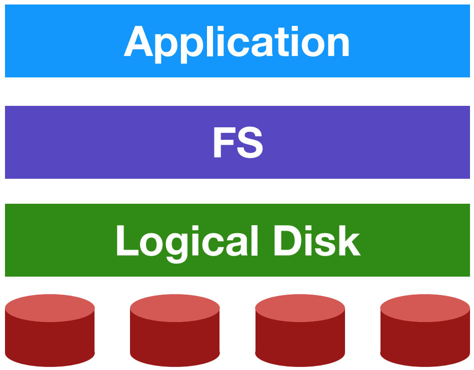



Application is smart, stores different files on different file systems.

JBOD: **J**ust a **B**unch **O**f **D**isks

# SOLUTION 2: RAID

Build logical disk from many physical disks.




Logical disk gives

capacity,

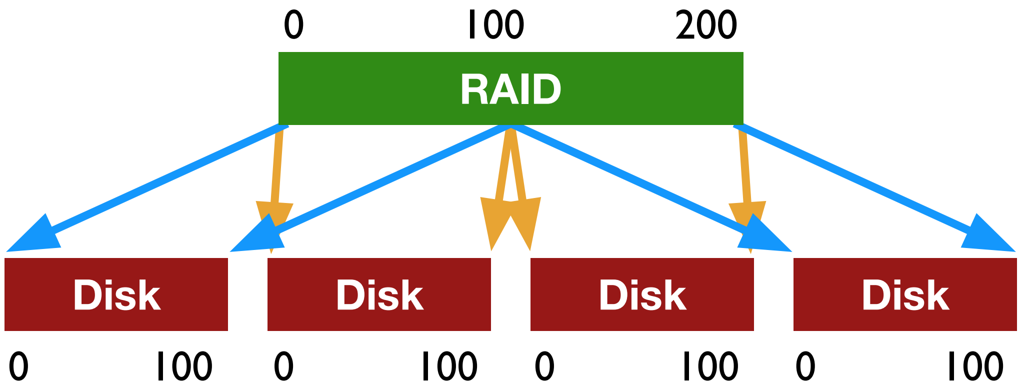
performance,

reliability

RAID: **R**edundant **A**rray of **I**nexpensive **D**isks

Transparency: no changes to the FS, host, Apps → ease of deployment

# General Strategy: Mapping, Redundancy



# MAPPING

How should we map logical block addresses to physical block addresses?

1) **Dynamic** mapping: use data structure (hash table, tree)

2) **Static** mapping: use simple math

# Fault Model Assumption

A disk can either work or not work

"Fail stop" model

Can detect when the disk is not working

No silent errors like LSE or block corruptions! (these are much harder to deal with)

lots of research on this topic

# WORKLOADS

- Reads
    - One operation
    - Steady-state I/O
        - Sequential
        - Random
- Writes
    - One operation
    - Steady-state I/O
        - Sequential
        - Random

# METRICS

**Capacity**: how much space can apps use?

**Reliability**: how many disks can we safely lose? (assume fail stop)

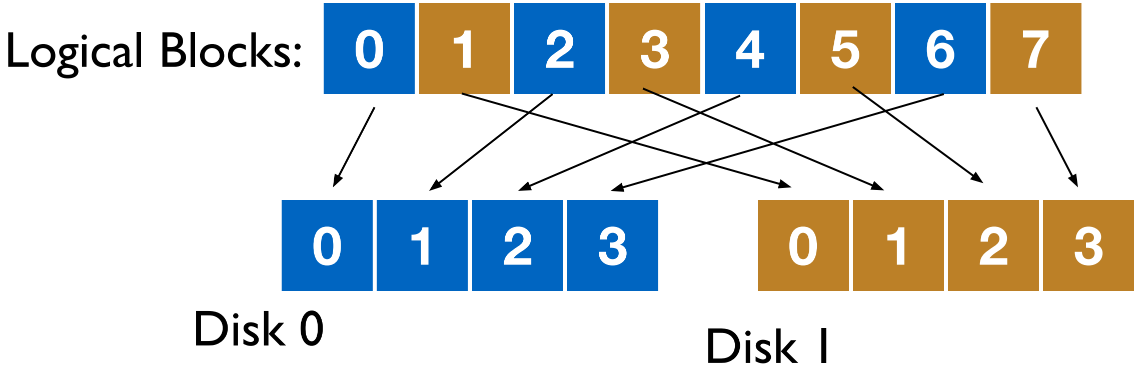
**Performance**: how long does each workload take? (latency, throughput)

Normalize each to characteristics of one disk

Different **RAID levels** make different trade-offs

# RAID-0: STRIPING

Optimize for capacity. No redundancy



# RAID 0: STRIPES AND CHUNK SIZE

Chunk size = 1

stripe:

| Disk 0 | Disk 1 | Disk 2 | Disk 4 |
|---|---|---|---|
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 |

assume chunk size of 1 for this lecture

Chunk size = 2

| Disk 0 | Disk 1 | Disk 2 | Disk 4 |
|---|---|---|---|
| 0 | 2 | 4 | 6 |
| 1 | 3 | 5 | 7 |
| 8 | 10 | 12 | 14 |
| 9 | 11 | 13 | 15 |

# RAID-0: Analysis

What is the capacity?

How many disks can we safely lose?

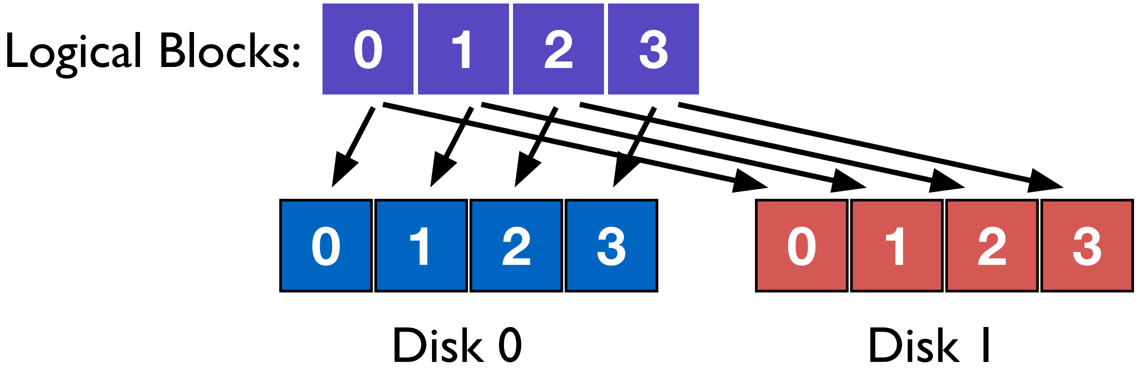
Latency (random)

Throughput (sequential, random)?

Discuss for two mins…

N := number of disks
C := capacity of 1 disk
S := sequential throughput of 1 disk
R := random throughput of 1 disk
D := latency of one small I/O operation

# RAID-1: MIRRORING




Keep two copies of all data.

# RAID-1 Layout: Mirroring

2 disks

| Disk 0 | Disk 1 |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

4 disks

| Disk 0 | Disk 1 | Disk 2 | Disk 4 |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 2 | 2 | 3 | 3 |
| 4 | 4 | 5 | 5 |
| 6 | 6 | 7 | 7 |

# RAID-1: ANALYSIS

What is the capacity?

How many disks can fail?

Latency (read, write)?

In practice, latency of random write would be slightly higher than the latency of random write in a single disk. Why?

N := number of disks
C := capacity of 1 disk
S := sequential throughput of 1 disk
R := random throughput of 1 disk
D := latency of one small I/O operation

| Disk 0 | Disk 1 |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

# RAID-1: Throughput

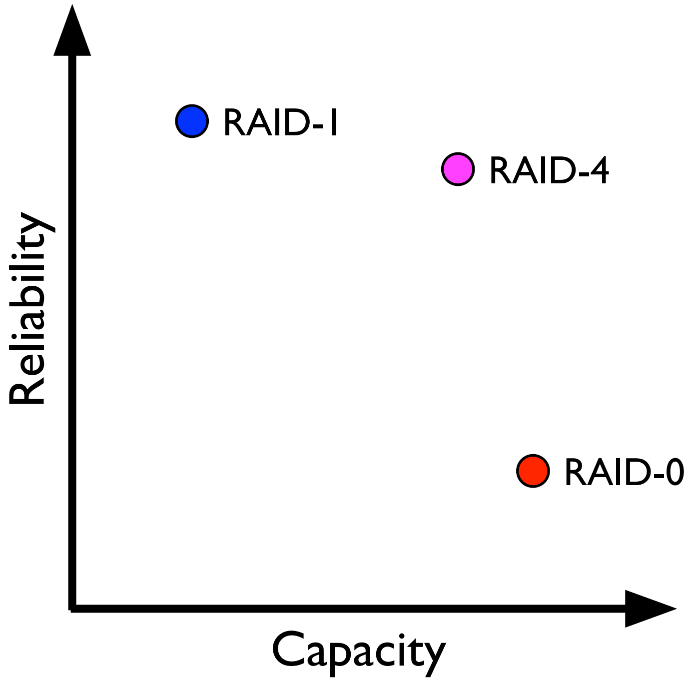
What is steady-state throughput for

- random reads?

- random writes?

- sequential writes?

- sequential reads?



RAID-1
RAID-4
RAID-0
Reliability
Capacity

# RAID-4 Strategy

Use **parity** disk

If an equation has N variables, and N-1 are known,
you can solve for the unknown.

Treat sectors across disks in a stripe as an equation.

Data on bad disk is like an unknown in the equation.

# RAID 4: EXAMPLE

| Disk 0 | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | P0 |
| 4 | 5 | 6 | 7 | P1 |
| 8 | 9 | 10 | 11 | P2 |
| 12 | 13 | 14 | 15 | P3 |

What functions can we use to compute parity?

| C0 | C1 | C2 | C3 | P |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | XOR(0,0,1,1) = 0 |
| 0 | 1 | 0 | 0 | XOR(0,1,0,0) = 1 |

# RAID-4: ANALYSIS

What is the capacity?

How many disks can fail?

Latency (read, write)?

N := number of disks
C := capacity of 1 disk
S := sequential throughput of 1 disk
R := random throughput of 1 disk
D := latency of one small I/O operation

# RAID-4: THROUGHPUT

What is steady-state throughput for

- sequential reads?
- sequential writes?
- random reads?
- random writes? (next page!)

| Disk0 | Disk1 | Disk2 | Disk3 | Disk4 |
|---|---|---|---|---|
| 3 | 0 | 1 | 2 | 6 |
| | | | | (parity) |

# RAID-4: ADDITIVE VS SUBTRACTIVE

| C0 | C1 | C2 | C3 | P0 |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | XOR(0,0,1,1) |

Additive Parity:
read all old blocks, recalculate parity block value

Subtractive Parity

$$P_{new} = (C_{old} \oplus C_{new}) \oplus P_{old}$$

| Disk 0 | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | P0 |
| *4 | 5 | 6 | 7 | +P1 |
| 8 | 9 | 10 | 11 | P2 |
| 12 | *13 | 14 | 15 | +P3 |

# RAID-5

| Disk0 | Disk1 | Disk2 | Disk3 | Disk4 |
|---|---|---|---|---|
| - | - | - | - | P |
| - | - | - | P | - |
| - | - | P | - | - |

...

Rotate parity across different disks

# RAID-5: ANALYSIS

What is capacity?

How many disks can fail?

Latency (read, write)?

| Disk0 | Disk1 | Disk2 | Disk3 | Disk4 |
|---|---|---|---|---|
| - | - | - | - | P |
| - | - | - | P | - |
| - | - | P | - | - |

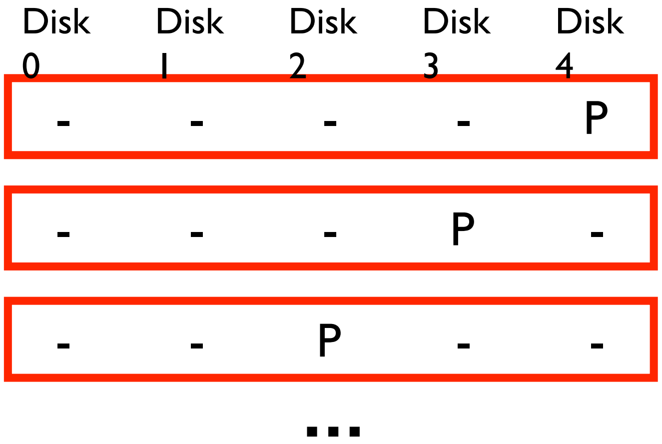
...

N := number of disks
C := capacity of 1 disk
S := sequential throughput of 1 disk
R := random throughput of 1 disk
D := latency of one small I/O operation

# RAID-5: THROUGHPUT

What is steady-state throughput for RAID-5?

- sequential reads?
- sequential writes?
- random reads?
- random writes? (next page!)



# RAID-5 RANDOM WRITES

| Disk 0 | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | P0 |
| 5 | 6 | 7 | P1 | 4 |
| 10 | 11 | P2 | 8 | 9 |
| 15 | P3 | 12 | 13 | 14 |
| P4 | 16 | 17 | 18 | 19 |

# RAID LEVEL COMPARISONS

| | Reliability | Capacity | Read latency | Write Latency | Seq Read | Seq Write | Rand Read | Rand Write |
|---|---|---|---|---|---|---|---|---|
| RAID-0 | 0 | C*N | D | D | N * S | N * S | N * R | N * R |
| RAID-1 | 1 | C*N/2 | D | D | N/2 * S | N/2 * S | N * R | N/2 * R |
| RAID-4 | 1 | (N-1) * C | D | 2D | (N-1)*S | (N-1)*S | (N-1)*R | R/2 |
| RAID-5 | 1 | (N-1) * C | D | 2D | (N-1)*S | (N-1)*S | N * R | N/4 * R |

# SUMMARY

RAID: a faster, larger, more reliable disk system

One logical disk built from many physical disk

Different mapping and redundancy schemes
Present different trade-offs

Next steps: Filesystems on Thu
P4a due on Friday!