[537] RAID

Chapter 38

Tyler Harter

10/29/14

Review Disks/Devices

Device Protocol Variants

Status checks: polling *vs.* interrupts

Data: PIO *vs.* DMA

Control: special instructions *vs.* memory-mapped I/O

Disks

Doing an I/O requires:
- seek
- rotate
- transfer

What is expensive?

Schedulers

Strategy: reorder requests to meet some goal
- performance (by making I/O sequential)
- fairness
- consistent latency

Usually in both OS and H/W.

CFQ (Linux Default)

Completely Fair Queueing.

Queue for each process.

Do weighted round-robin between queues, with slice time proportional to priority.

Optimize order within queue.

Yield slice only if idle for a given time (anticipation).

RAID

Only One Disk?

Sometimes we want many disks — why?

Only One Disk?

Sometimes we want many disks — why?

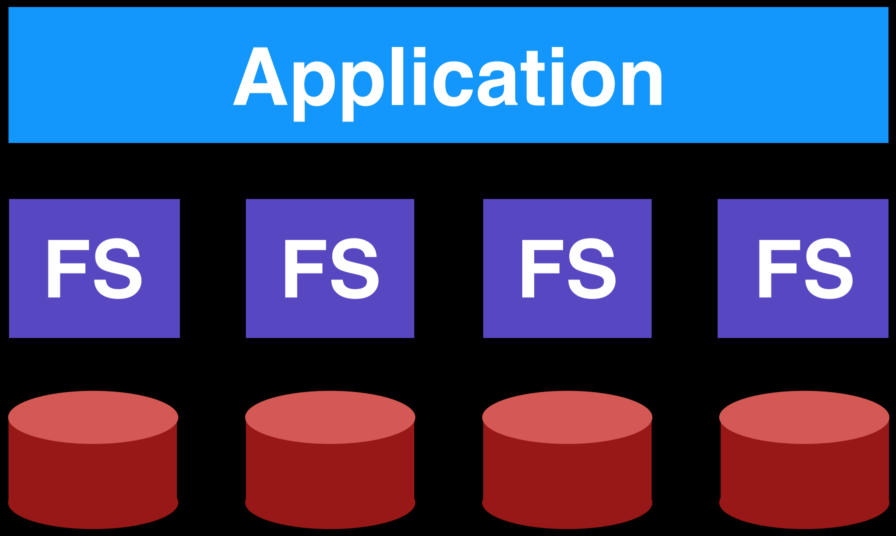
- capacity
- performance
- reliability

Only One Disk?

Sometimes we want many disks — why?
- capacity
- performance
- reliability

Challenge: most file systems work on only one disk.

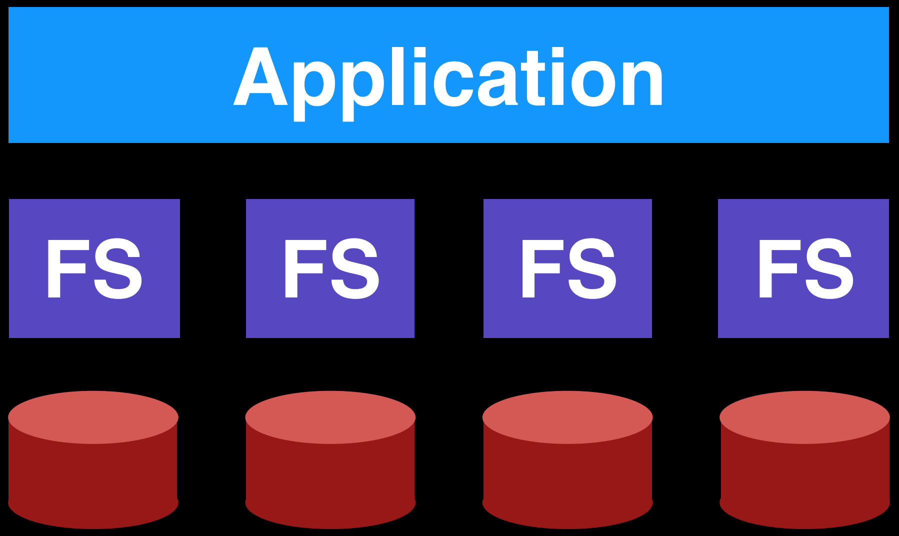
Solution 1: JBOD



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

Solution 1: JBOD

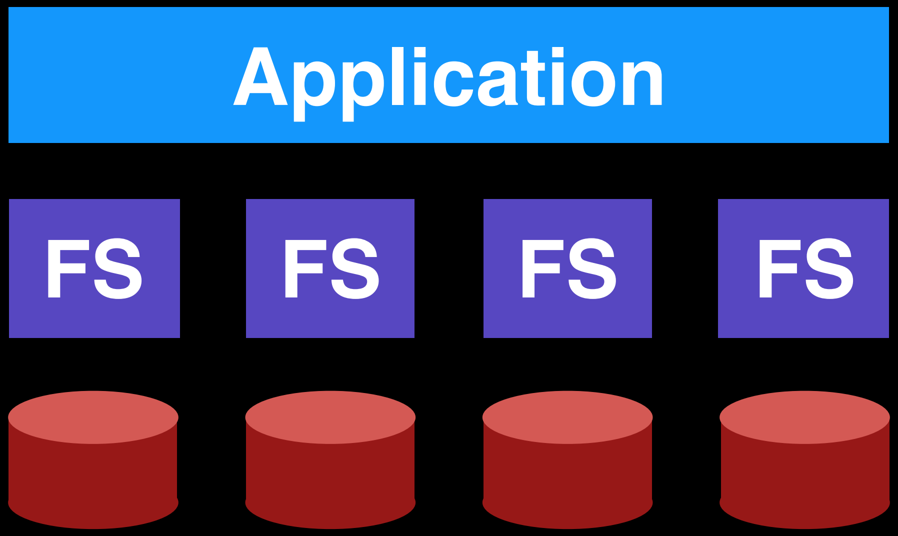
JBOD: **J**ust a **B**unch **O**f **D**isks (seriously!)



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

Solution 1: JBOD

JBOD: **J**ust a **B**unch **O**f **D**isks (seriously!)

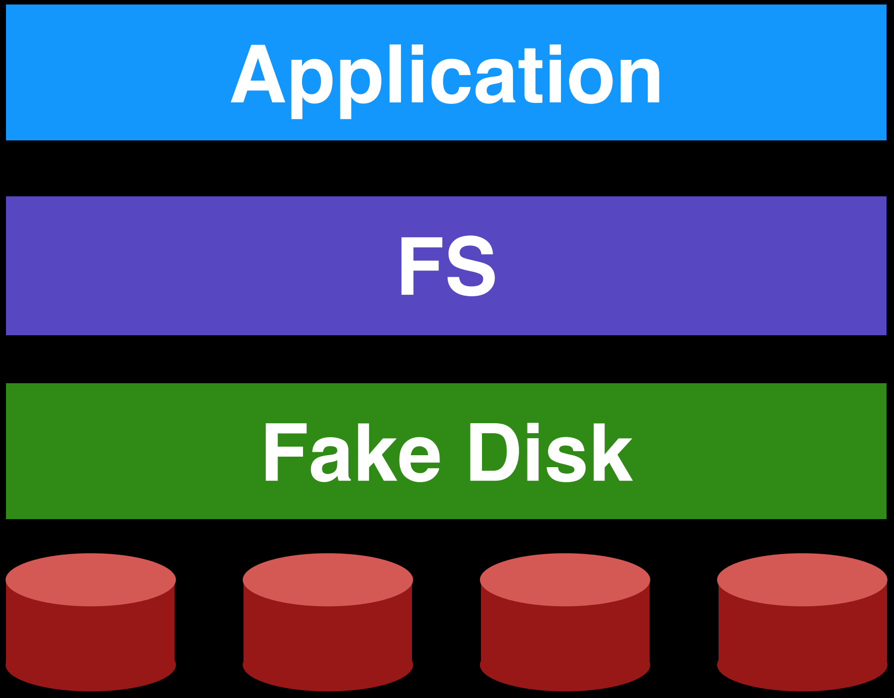


alternatives?

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

Solution 2: RAID

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

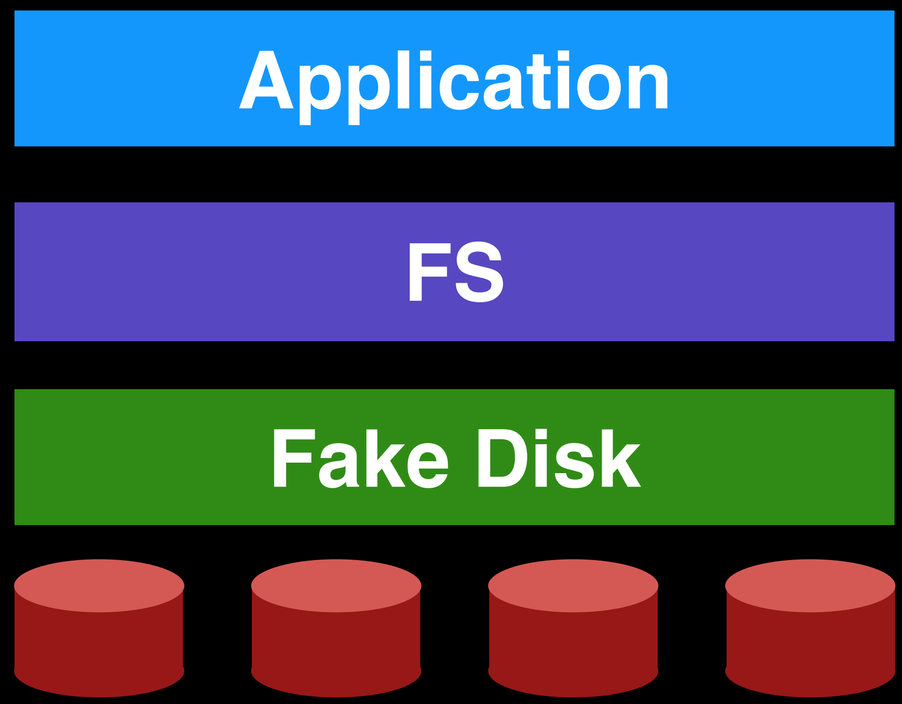


Build logical disk from many physical disks.

Solution 2: RAID

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

RAID is:
- transparent
- deployable

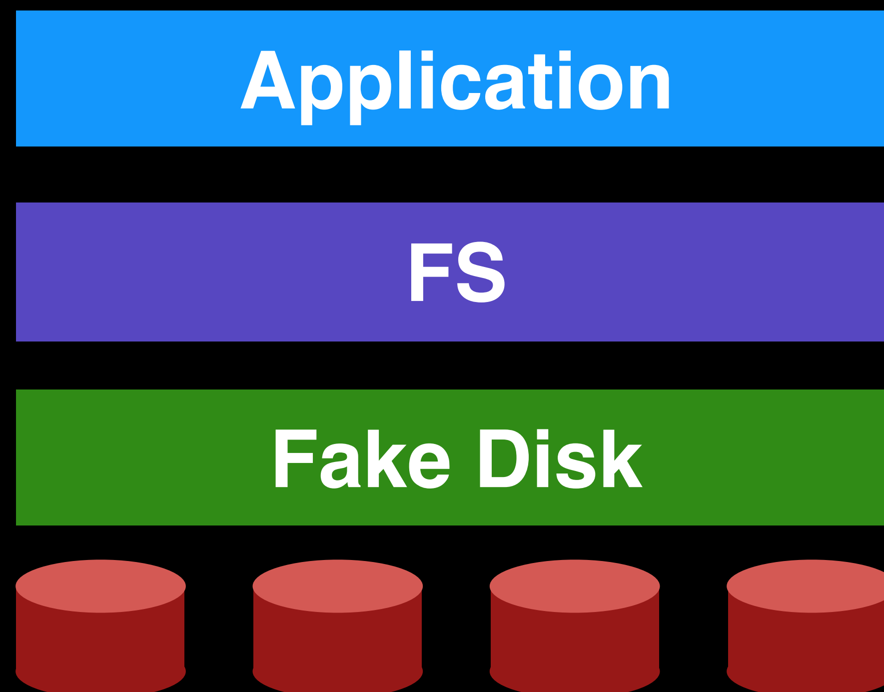


Build logical disk from many virtual disks.

Solution 2: RAID

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

RAID is:
- transparent
- deployable



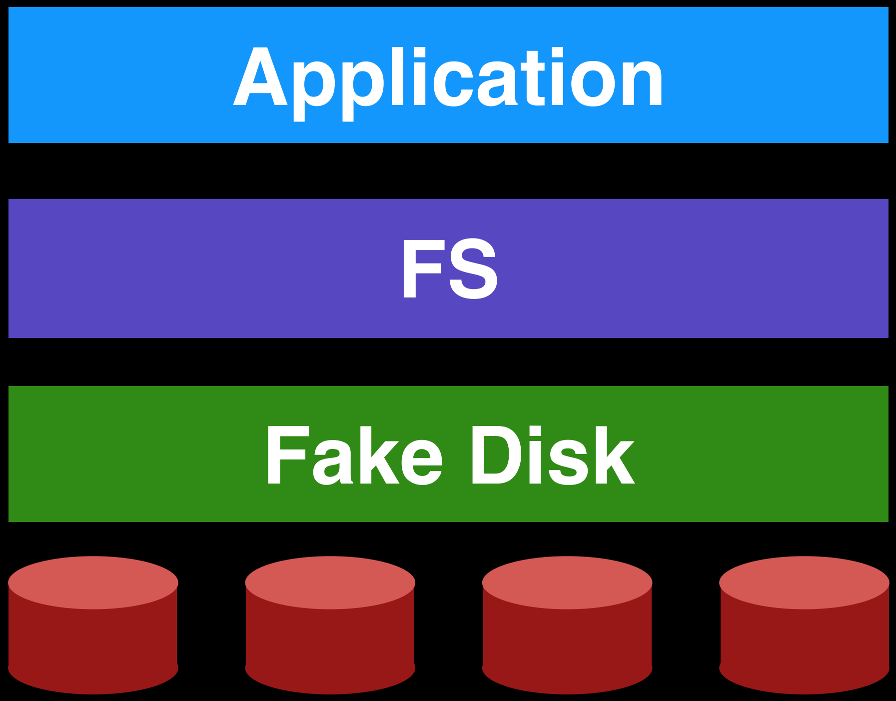
Logical disk gives
- capacity
- performance
-

Build logical disk from many virtual disks.

Solution 2: RAID

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

RAID is:
- transparent
- deployable



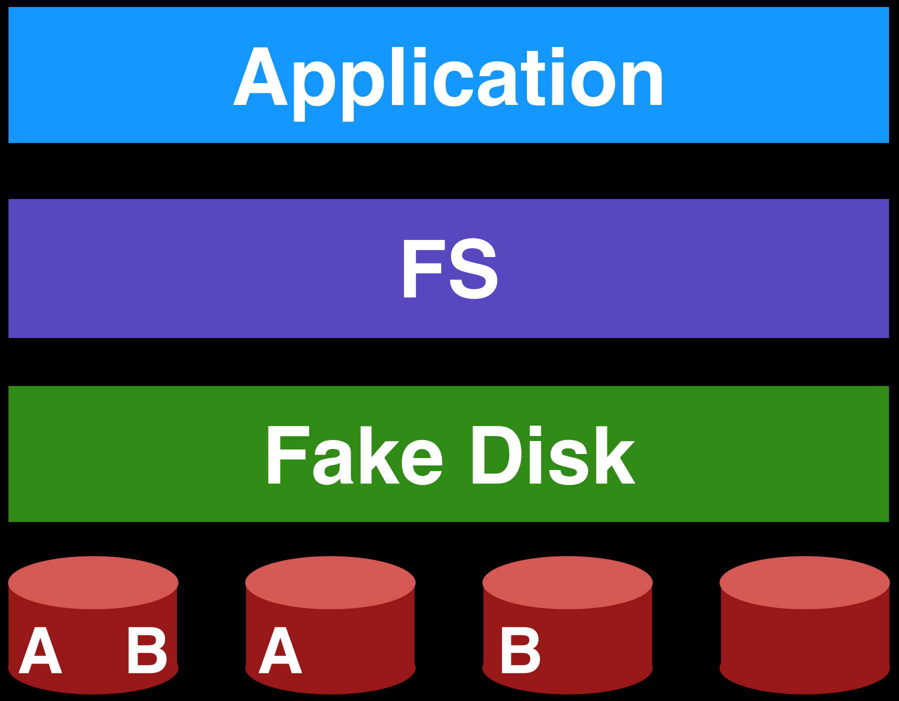
Logical disk gives
- capacity
- performance
- reliability?

Build logical disk from many virtual disks.

Solution 2: RAID

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

RAID is:
- transparent
- deployable



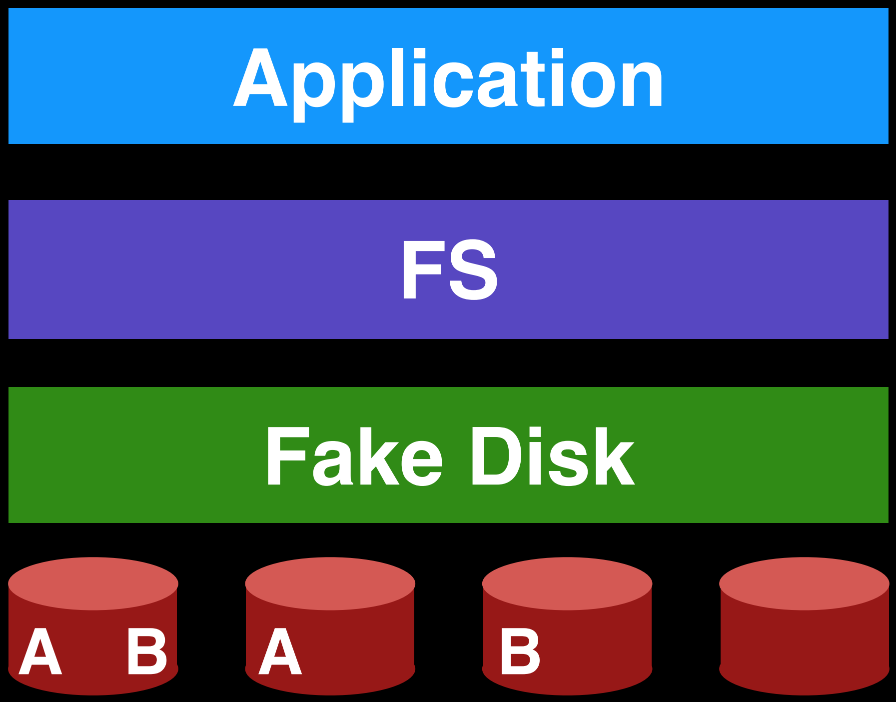
Logical disk gives
- capacity
- performance
- reliability?

Build logical disk from many virtual disks.

Solution 2: RAID

RAID: **R**edundant **A**rray of <u>**I**nexpensive **D**isks</u>

RAID is:
- transparent
- deployable



Logical disk gives
- capacity
- performance
- reliability?

Build logical disk from many virtual disks.

Why *Inexpensive* Disks?

Economies of scale! Cheap disks are popular.

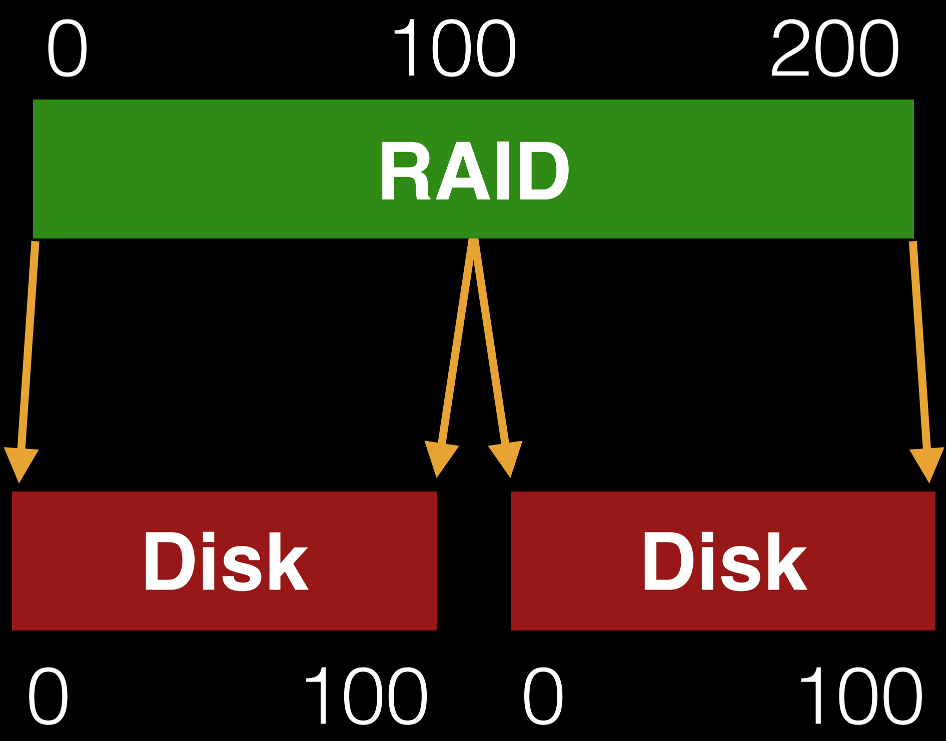
You can often get many commodity H/W components for the same price as a few expensive components.

Strategy: write S/W to build high-quality logical devices from many cheap devices.

Alternative to RAID: buy an expensive, high-end disk.

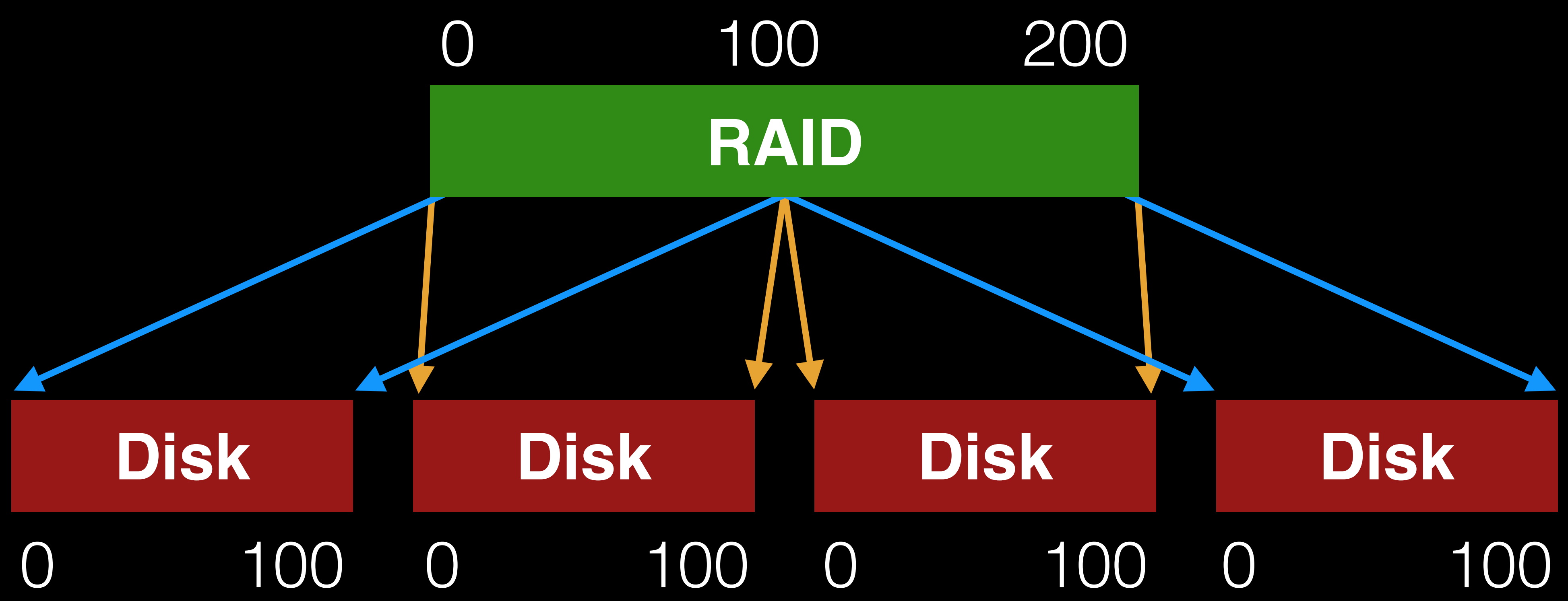
General Strategy

Build fast, large disk from smaller ones.



General Strategy

Add even more disks for reliability.



Mapping

How should we map logical to physical addresses?

How is this problem similar to virtual memory?

Mapping

How should we map logical to physical addresses?

How is this problem similar to virtual memory?

Dynamic mapping: use data structure (hash table, tree)
- paging

Static mapping: use math
- RAID

Mapping

How should we map logical to physical addresses?

How is this problem similar to virtual memory?

Dynamic mapping: use data structure (hash table, tree)
- paging

Static mapping: use math
- RAID

RAID volume is fixed-sized, dense

Redundancy

Redundancy: how many copies?

System engineers are always trying to increase or decrease redundancy.

Increase: replication (e.g., RAID)
Decrease: deduplication (e.g., code sharing)

Redundancy

Redundancy: how many copies?

System engineers are always trying to increase or decrease redundancy.

Increase: replication (e.g., RAID)
Decrease: deduplication (e.g., code sharing)

Why are we so fickle?

Redundancy

Increase: improves reliability
Decrease: improves space efficiency

One strategy: reduce redundancy as much is possible. Then add back just the right amount.

Redundancy

Increase: improves reliability
Decrease: improves space efficiency

One strategy: reduce redundancy as much is possible. Then add back just the right amount.

Take course on channel/source coding.
(e.g., ECE 729)

RAID Analysis

Reasoning About RAID

Workload: types of reads/writes issued by app

RAID: system for mapping logical to physical addrs

Metric: capacity, reliability, performance

RAID "algebra", given 2 variables, find the 3rd:

$$f(\mathbf{W}, \mathbf{R}) = \mathbf{M}$$

Reasoning About RAID

Workload: types of reads/writes issued by app

RAID: system for mapping logical to physical addrs

Metric: capacity, reliability, performance

RAID "algebra", given 2 variables, find the 3rd:

$$f(\mathbf{W}, \mathbf{R}) = \mathbf{M}$$

RAID Decisions

Which logical blocks map to which physical blocks?

How do we use extra physical blocks (if any)?

Reasoning About RAID

Workload: types of reads/writes issued by app

RAID: system for mapping logical to physical addrs

Metric: capacity, reliability, performance

RAID "algebra", given 2 variables, find the 3rd:

$$f(W, R) = M$$

Workloads

Reads
Writes

Workloads

- Reads
    - One operation
    - Steady I/O
- Writes
    - One operation
    - Steady I/O

Workloads

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

Reasoning About RAID

Workload: types of reads/writes issued by app

RAID: system for mapping logical to physical addrs

Metric: capacity, reliability, performance

RAID "algebra", given 2 variables, find the 3rd:

$$f(\mathbf{W}, \mathbf{R}) = \mathbf{M}$$

Metrics

Capacity: how much space can apps use?

Reliability: how many disks can we safely lose?

Performance: how long does each workload take?

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?

Metrics

Capacity: how much space can apps use?

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

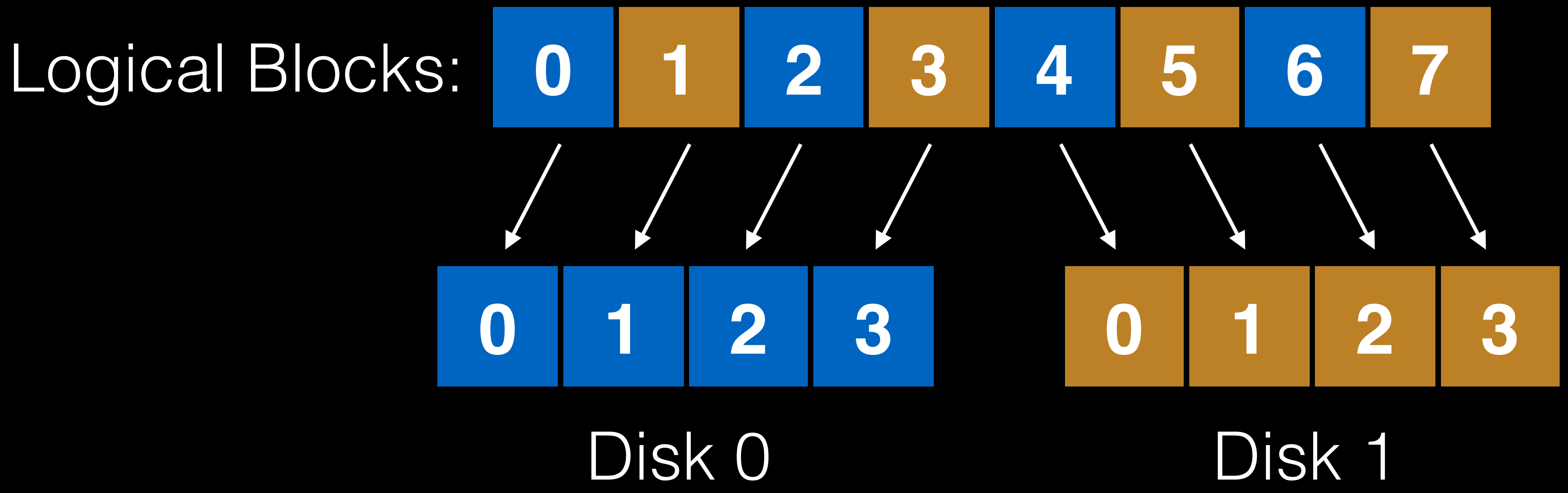
Performance: how long does each workload take?

Normalize each to characteristics of one disk
(see definitions on worksheet).

RAID-0

RAID-0: Striping

Optimize for capacity. No redundancy (weird name).



Another View

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

4 disks

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

4 disks

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

How to Map

Given logical address A, find:

Disk = …

Offset = …

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

How to Map

Given logical address A, find:
Disk = A % disk_count
Offset = A / disk_count

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

Chunk Size = 1

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

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

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

Chunk Size = 2

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

Chunk Size = 2

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

We'll assume chunk size of 1 for today.
Sizes of 64KB are typical in deployment.

Chunk Size = 2

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

When are small chunks better?
When are big ones better?

(Problem C)

RAID-0: Analysis

0a) What is capacity?

0b) How many disks can fail?

0c) Throughput?

0d) Latency?

RAID-0: Analysis

0a) What is capacity? **N * C**

0b) How many disks can fail?

0c) Throughput?

0d) Latency?

RAID-0: Analysis

0a) What is capacity? **N * C**

0b) How many disks can fail? **0**

0c) Throughput?

0d) Latency?

RAID-0: Analysis

0a) What is capacity? **N * C**

0b) How many disks can fail? **0**

0c) Throughput? **N*S** (i,ii) and **N*R** (iii,iv)

0d) Latency?

RAID-0: Analysis

0a) What is capacity? **N * C**

0b) How many disks can fail? **0**

0c) Throughput? **N*S** (i,ii) and **N*R** (iii,iv)

0d) Latency? **D** (i,ii)

RAID-0: Analysis

0a) What is capacity? **N * C**

0b) How many disks can fail? **0**

0c) Throughput? **N*S** (i,ii) and **N*R** (iii,iv)

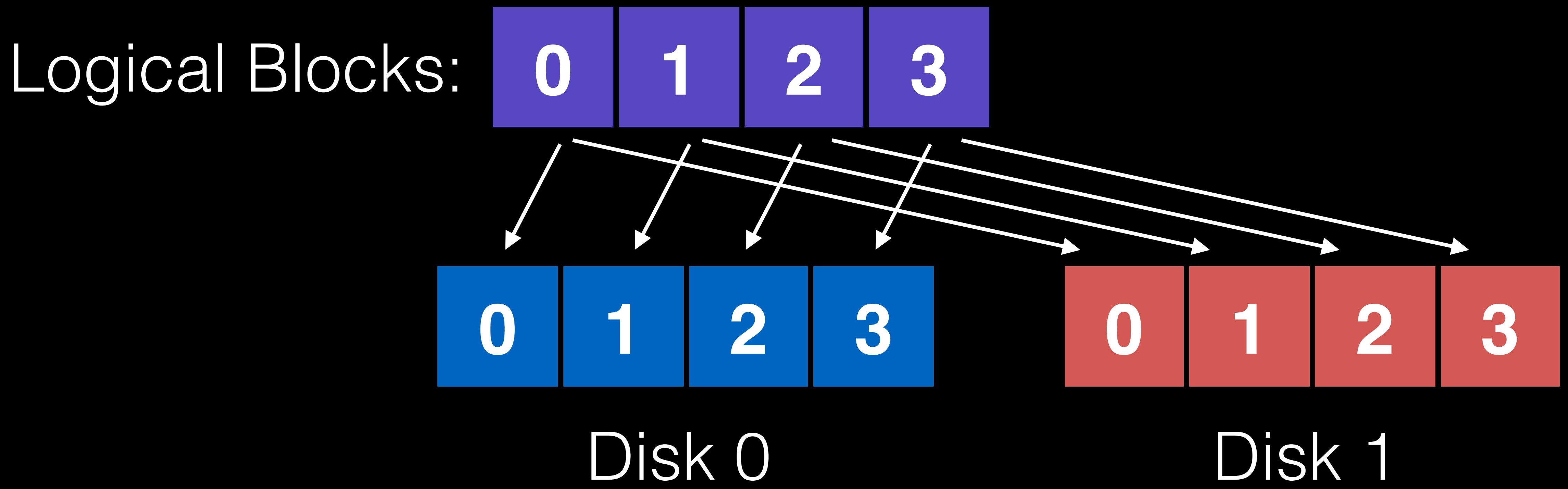
0d) Latency? **D** (i,ii)

Buying more disks improves throughput, but not latency!

RAID-1

RAID-1: Mirroring

Keep two copies of all data.



Assumptions

Assume disks are **fail-stop**.
- they work or they don't
- we know when they don't

Tougher Errors:
- latent sector errors
- silent data corruption

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

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

How many disks can fail?

RAID-1: Analysis

0a) What is capacity? **N/2 * C**

0b) How many disks can fail? **1 (or maybe N / 2)**

0c) Throughput? **???**

0d) Latency? **D** (i,ii)

RAID-1: Throughput

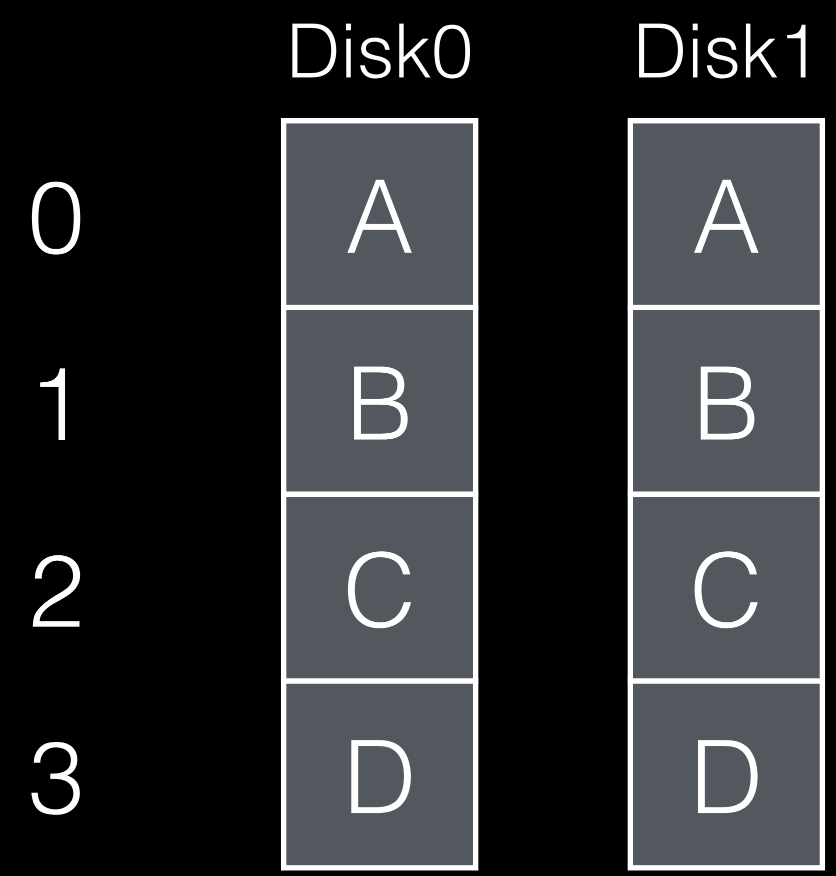
What is steady-state throughput for
- sequential reads?
- sequential writes?
- random reads?
- random writes?

RAID-1: Throughput

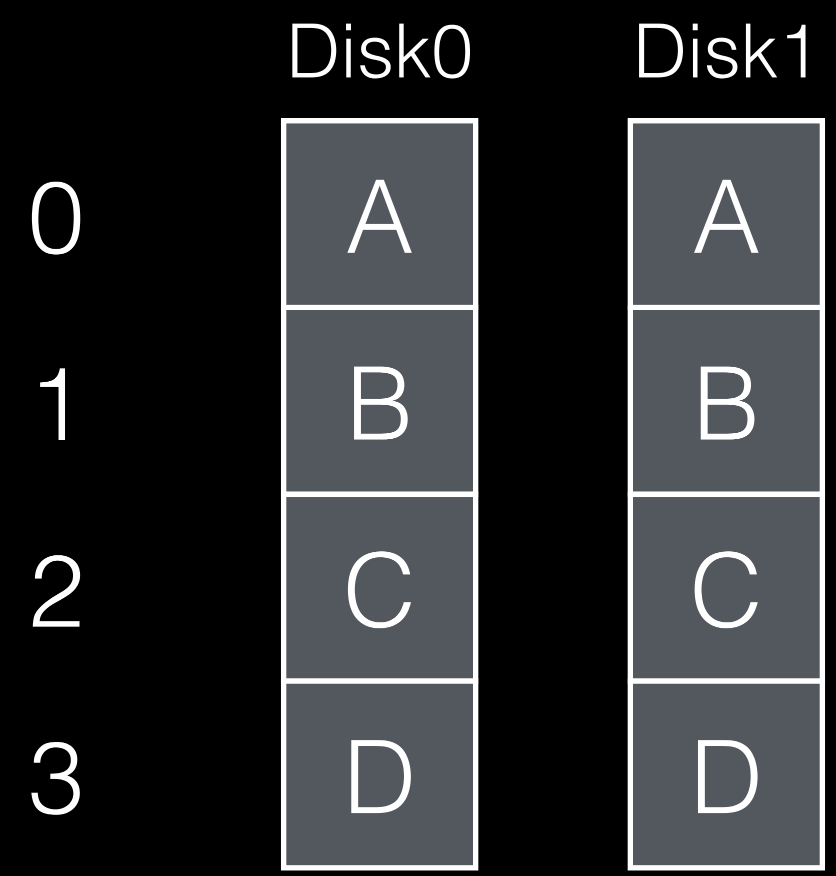
What is steady-state throughput for

- sequential reads? **N/2 * S**
- sequential writes? **N/2 * S**
- random reads? **N * R**
- random writes? **N/2 * R**

Crashes

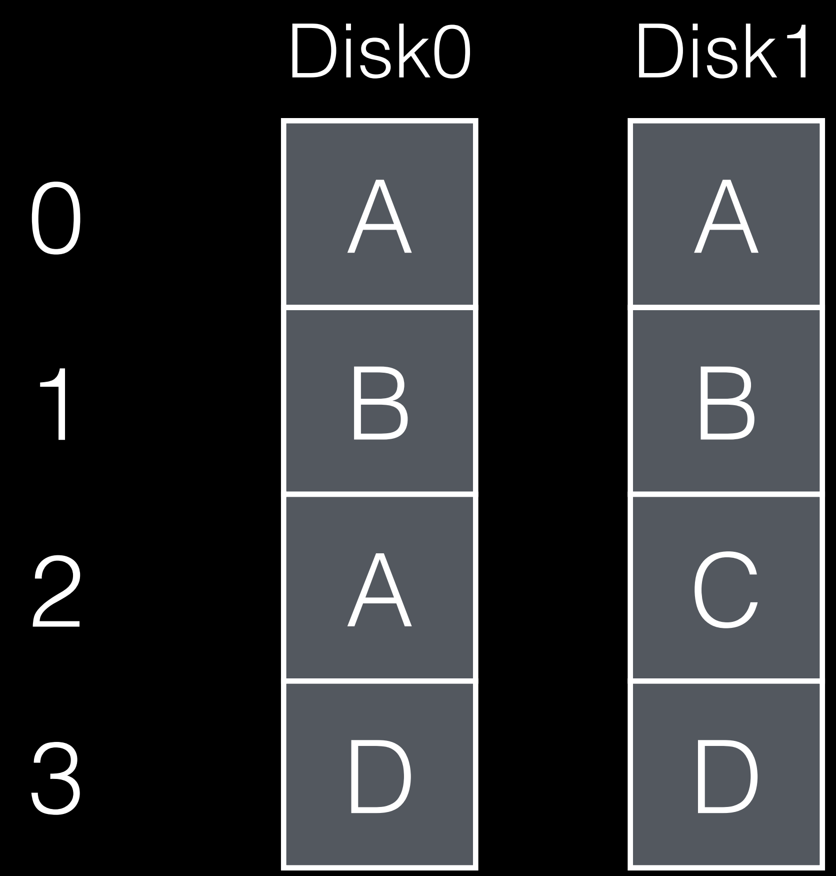


Crashes



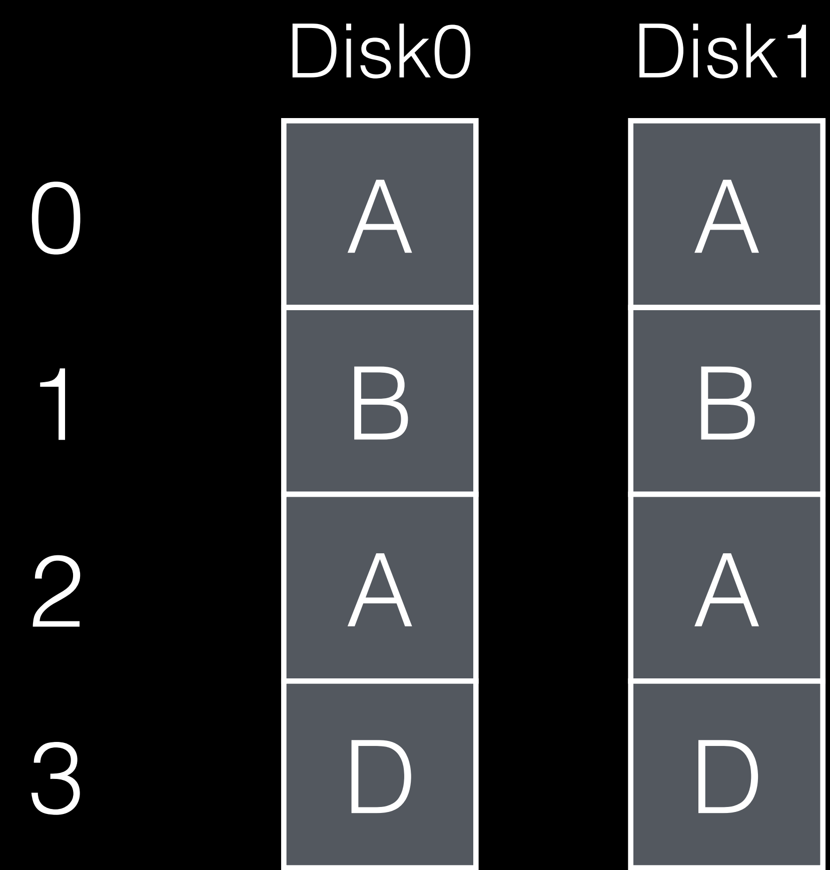
write(A) to 2

Crashes



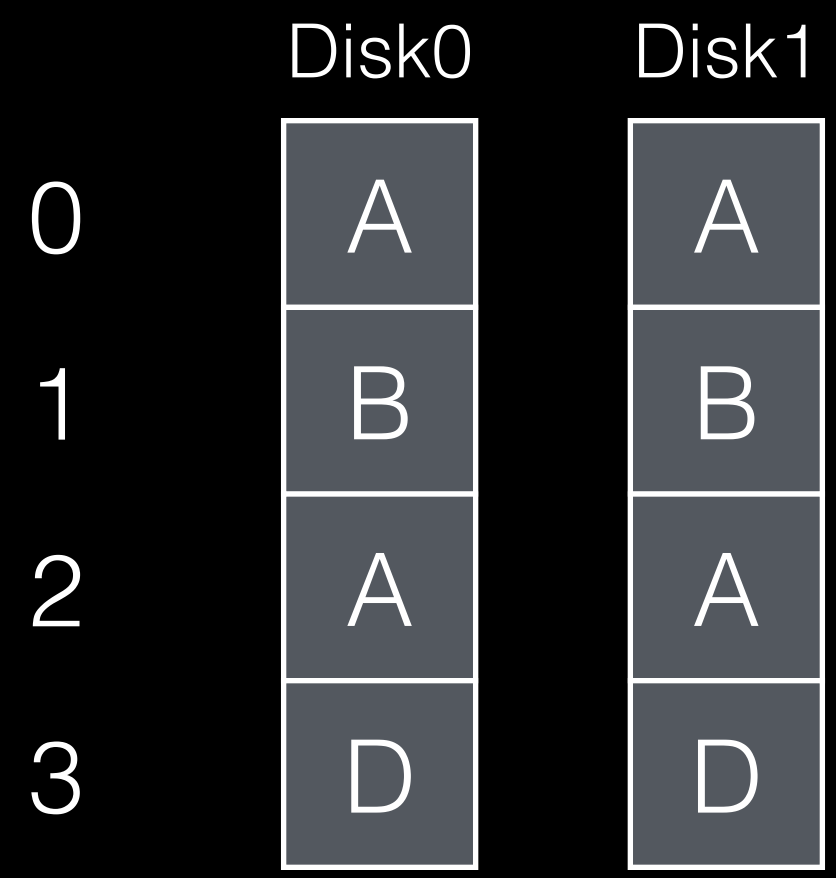
write(A) to 2

Crashes

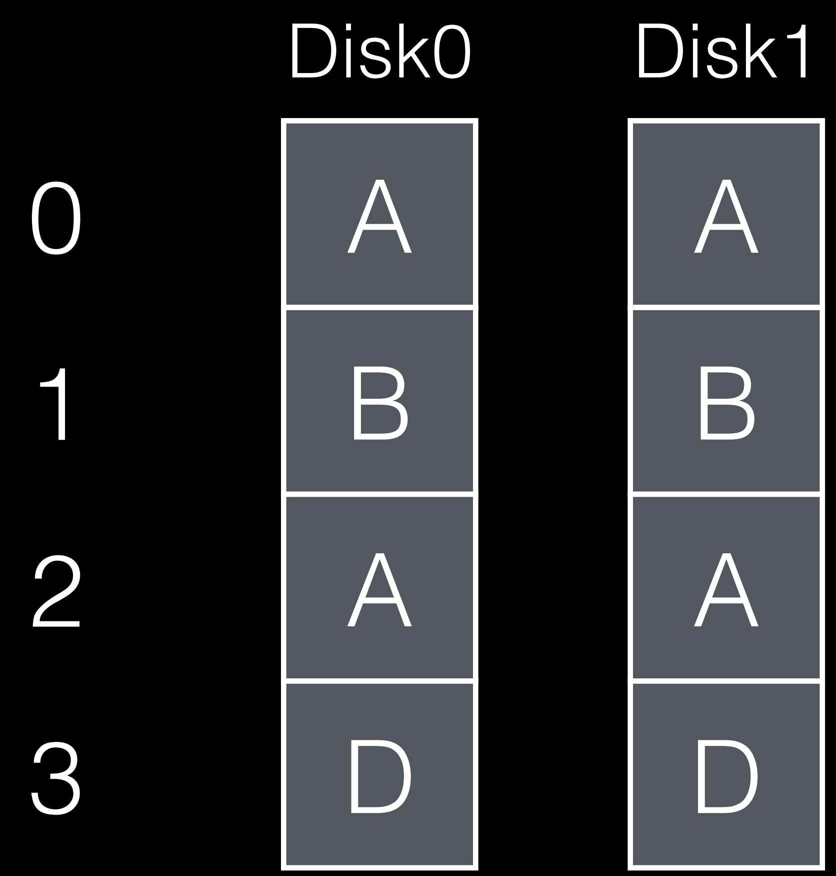


write(A) to 2

Crashes

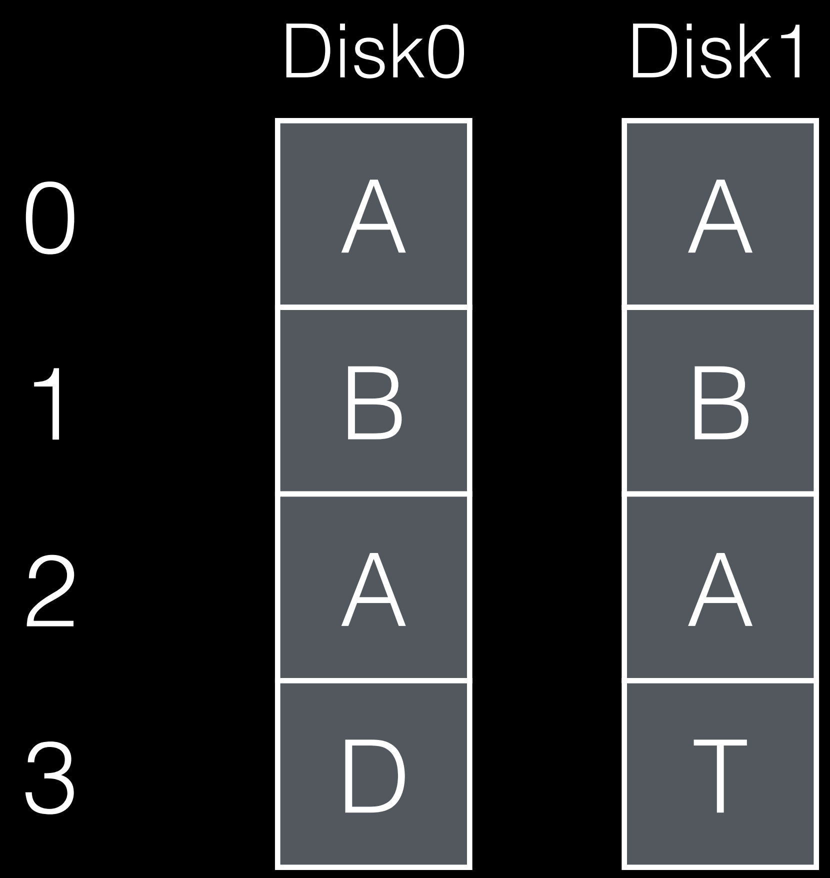


Crashes



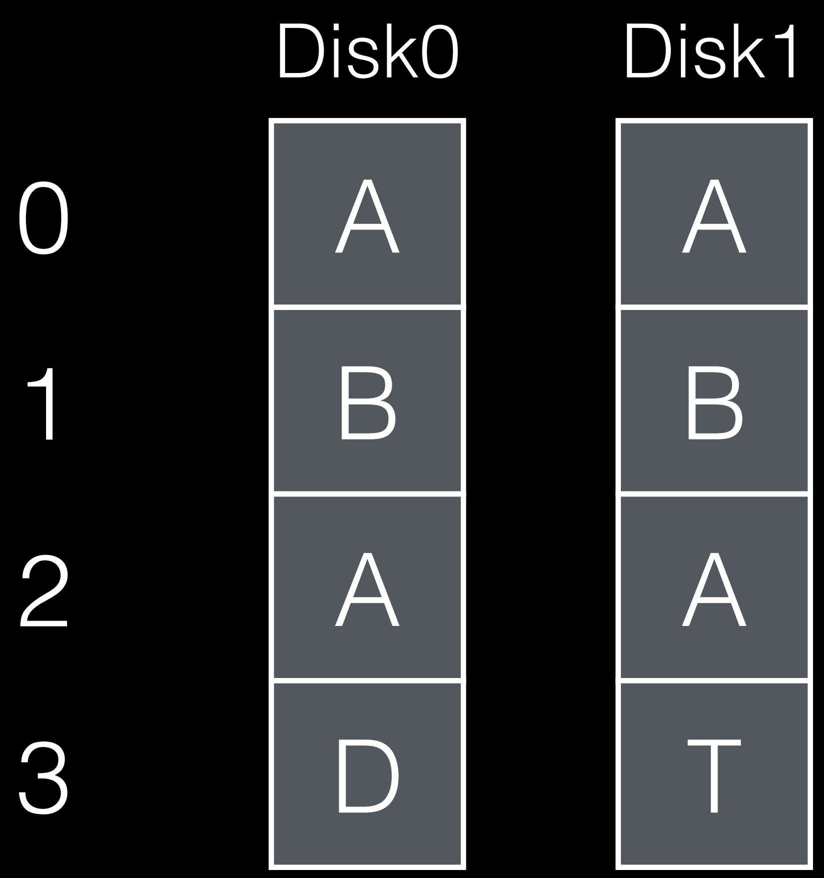
write(T) to 3

Crashes



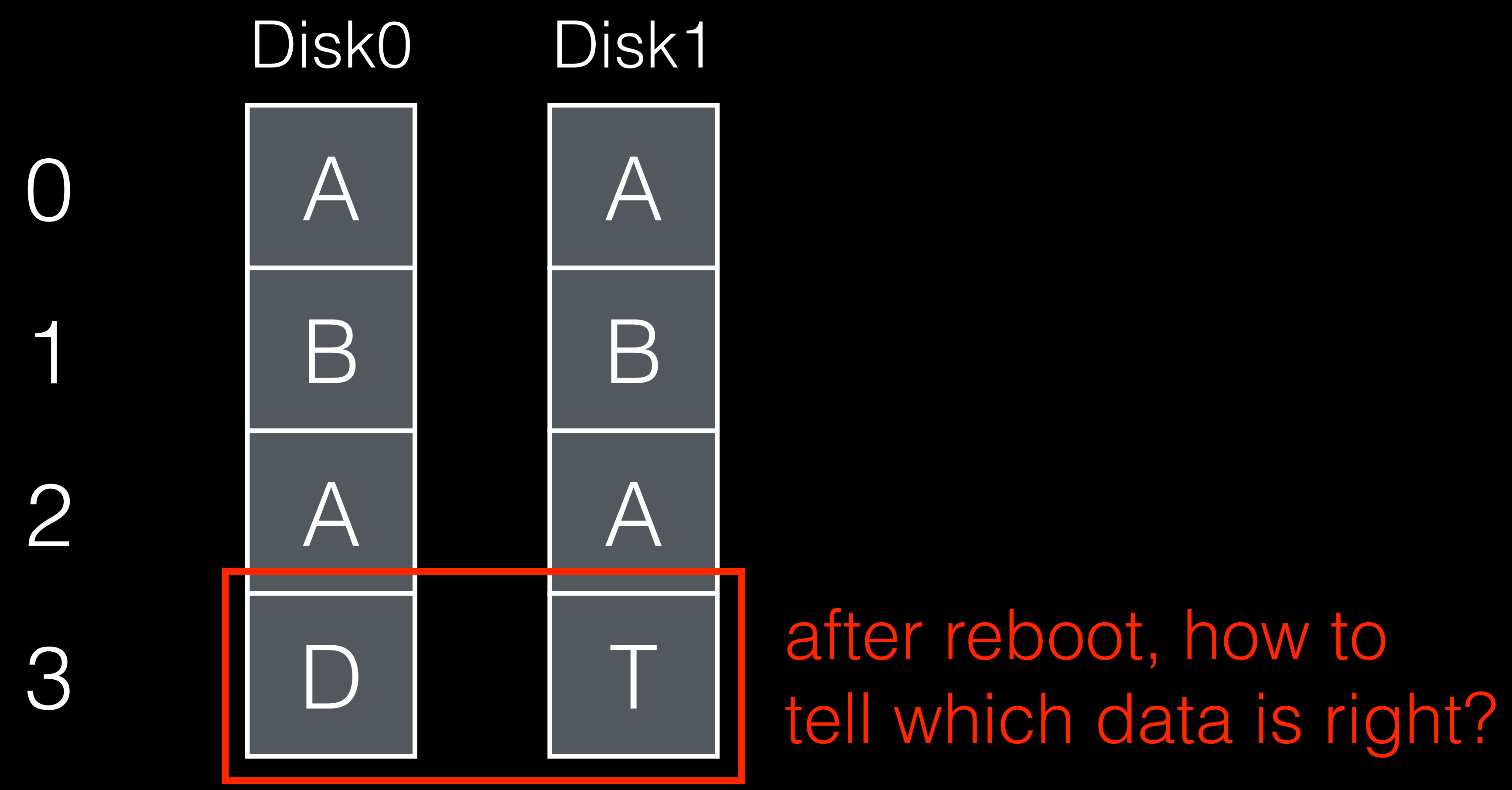
write(T) to 3

Crashes



CRASH!!!

Crashes


Disk0
Disk1
0
1
2
3
A
A
B
B
A
A
D
T
after reboot, how to
tell which data is right?

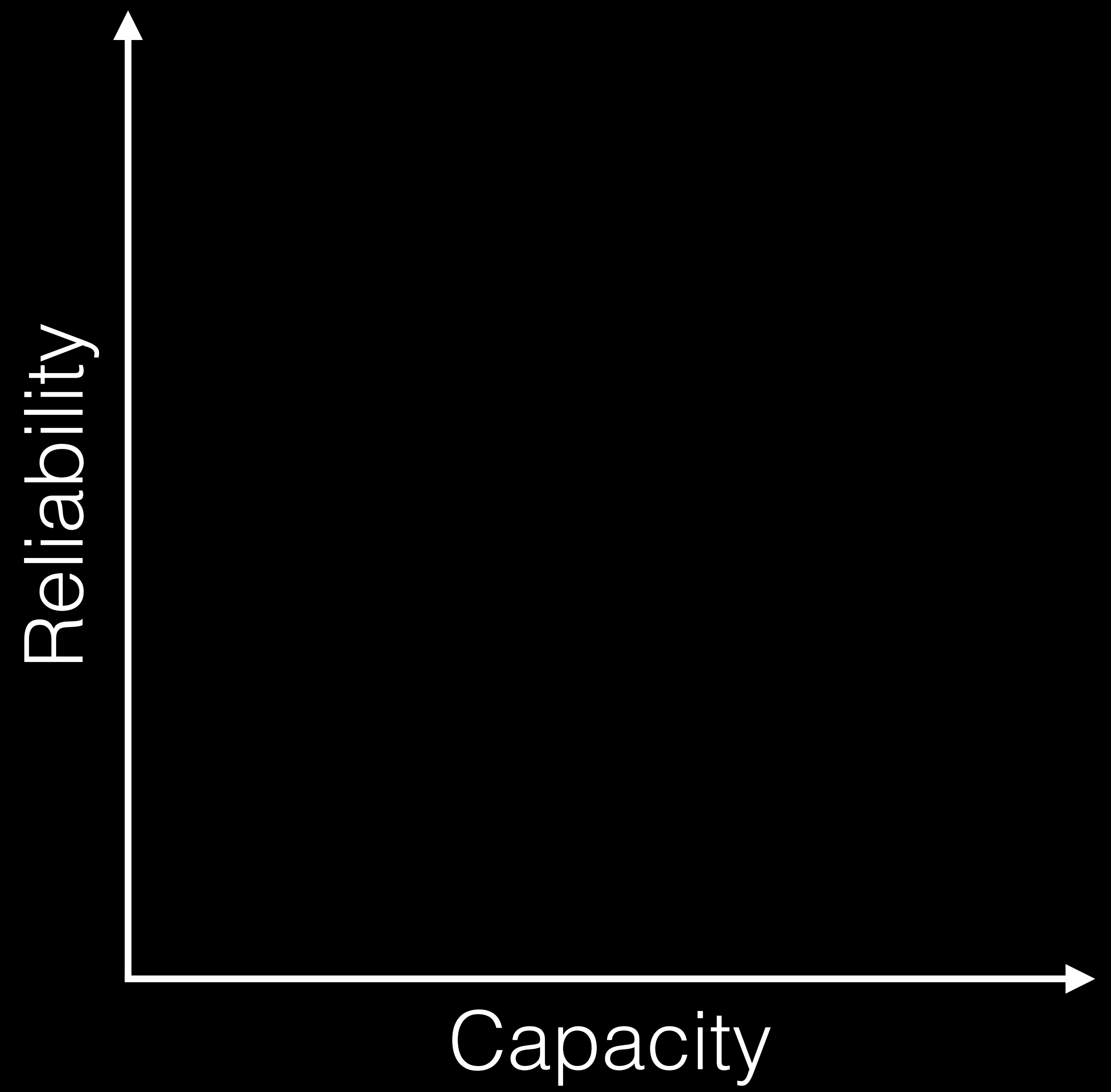
H/W Solution

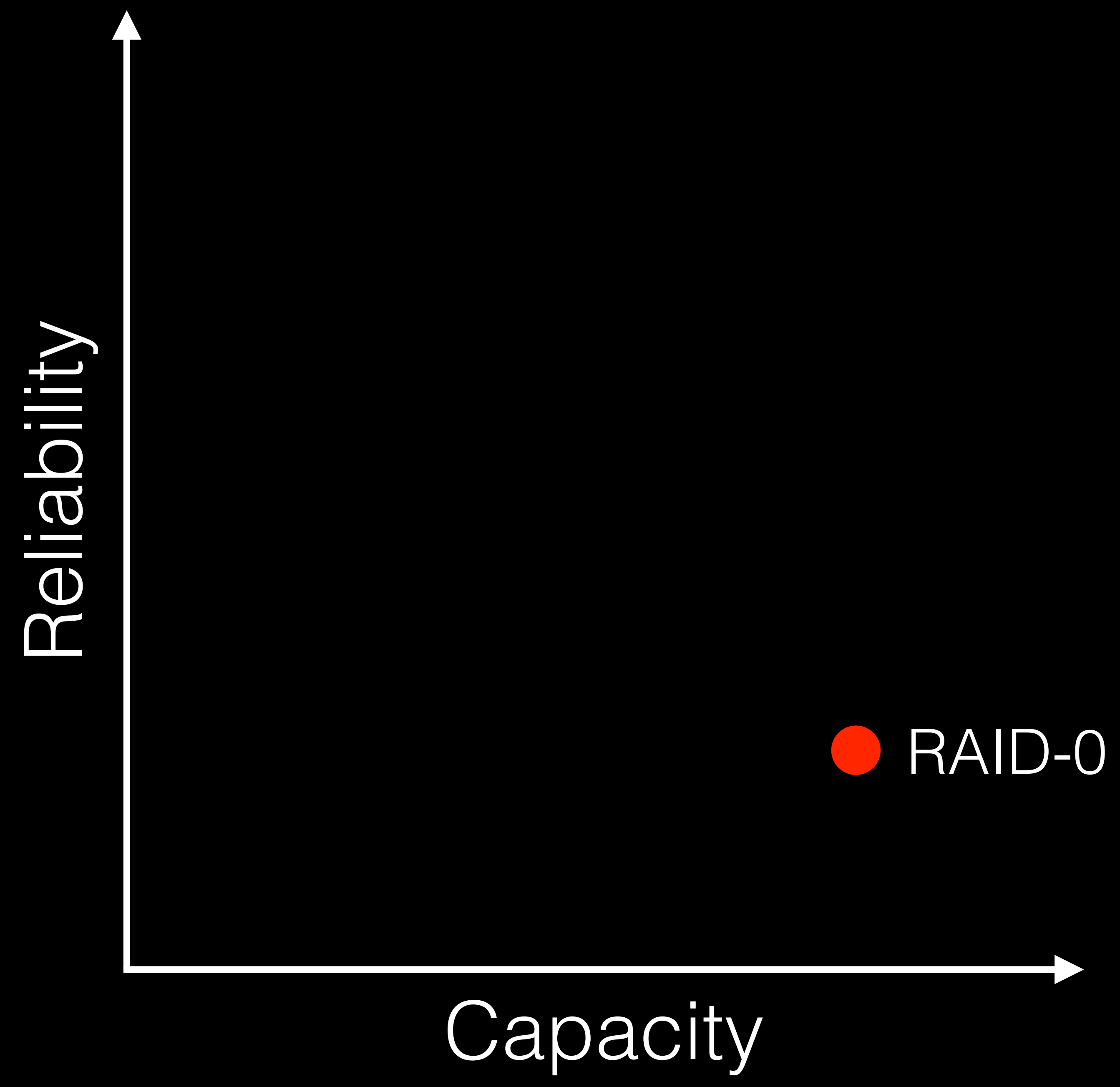
Problem: Consistent-Update Problem

Use non-volatile RAM in RAID controller.

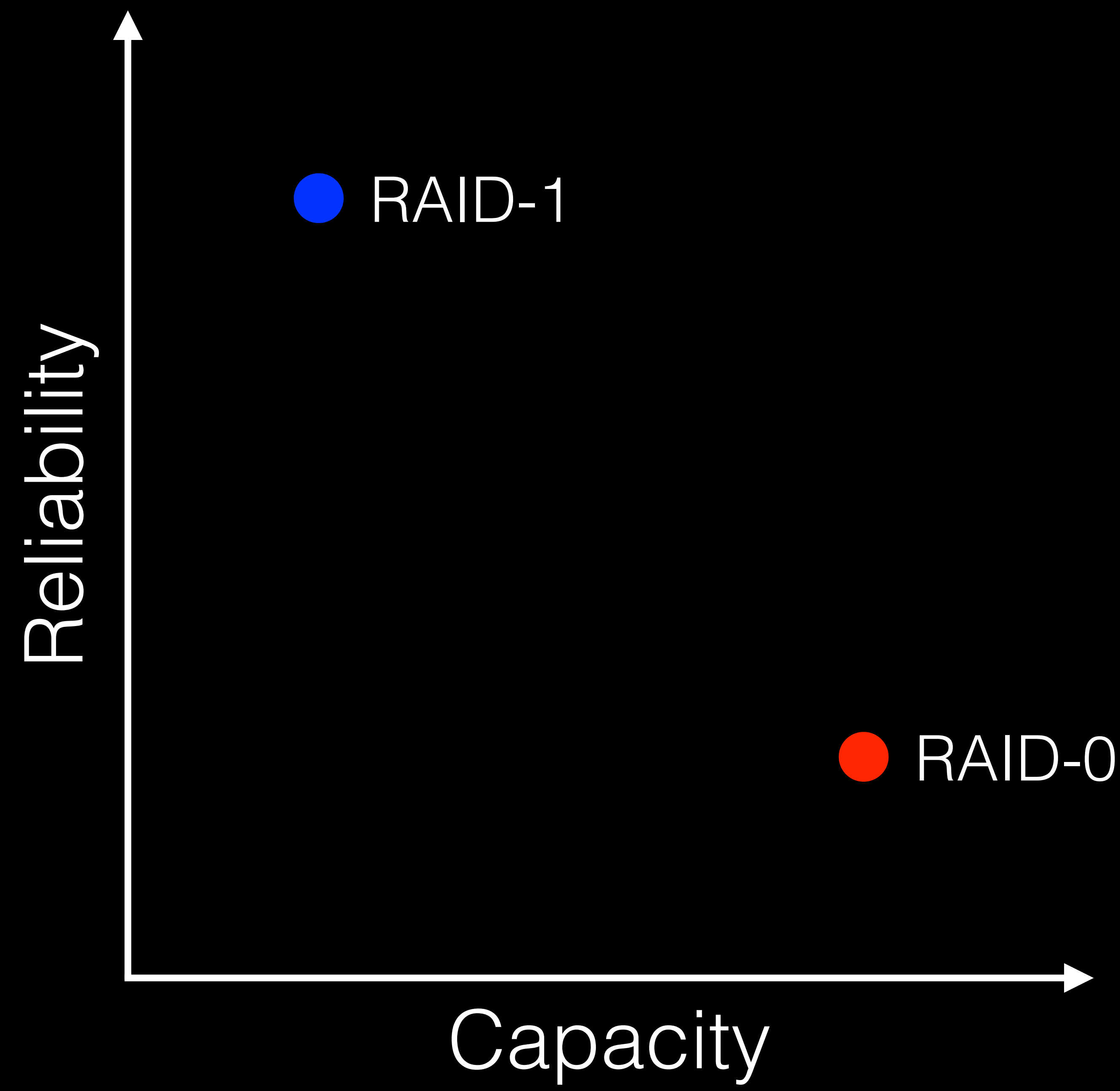
Software RAID controllers (e.g., Linux md) don't have this option.

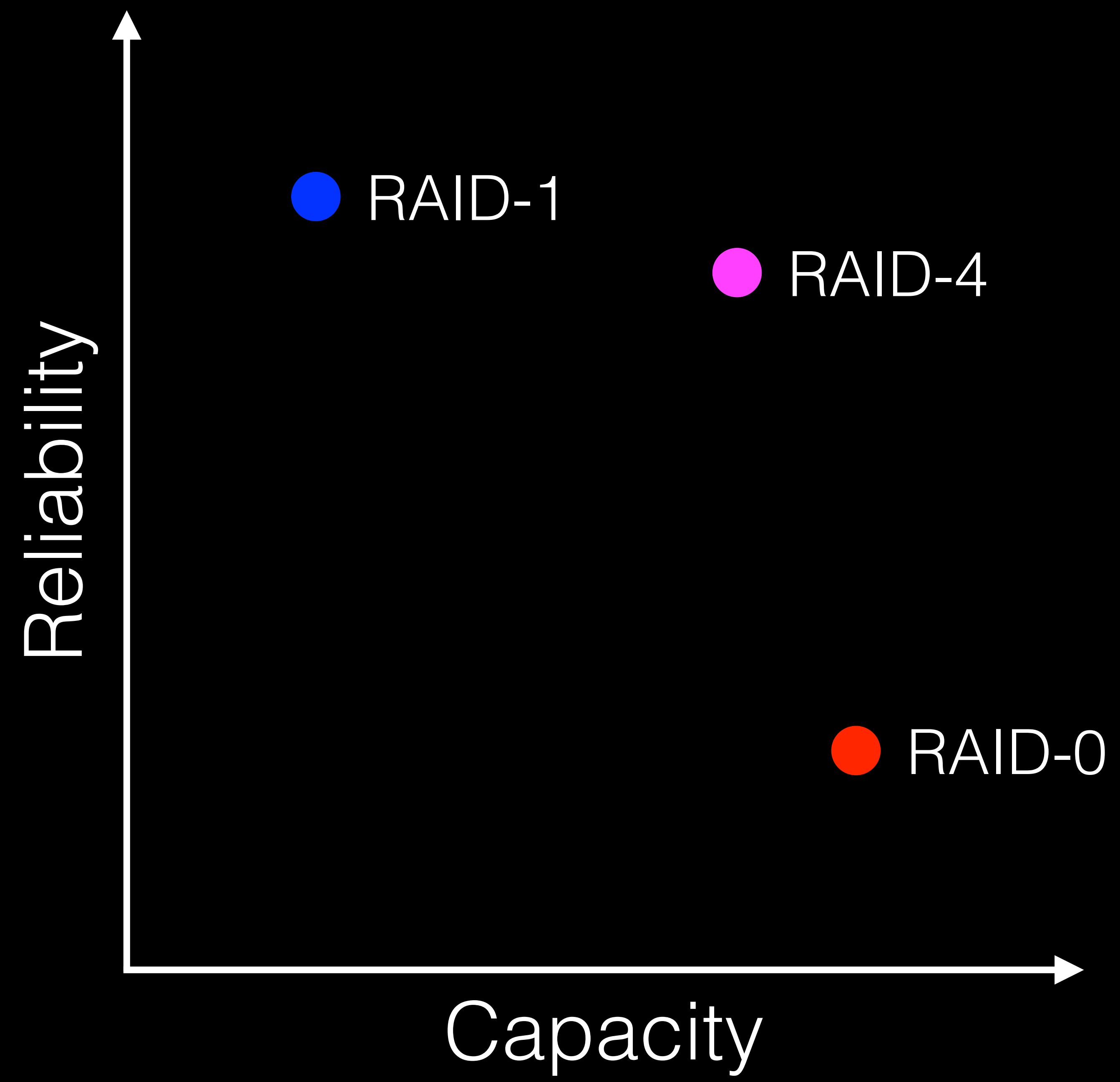
RAID-4


Reliability
Capacity



Reliability
RAID-0
Capacity


RAID-1
RAID-0
Reliability
Capacity


RAID-1
RAID-4
Reliability
RAID-0
Capacity

Strategy

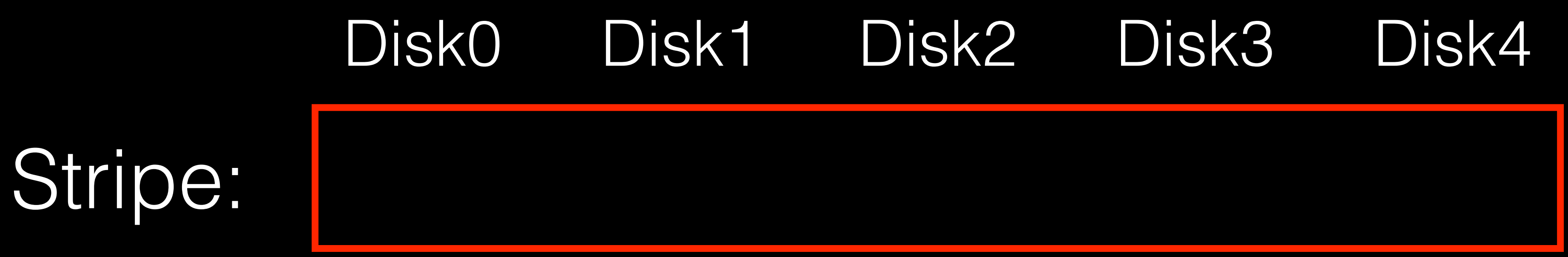
Use parity disk.

In algebra, if an equation has N variables, and N-1 are know, you can often solve for the unknown.

Treat the sectors across disks in a stripe as an equation.

A bad disk is like an unknown in the equation.

Example



Example

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	5	3	0	1	
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	5	3	0	1	9
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	5	X	0	1	9
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	5	3	0	1	9
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	2	1	1	X	5
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	2	1	1	1	5
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	3	0	1	2	X
					(parity)

Example

	Disk0	Disk1	Disk2	Disk3	Disk4
Stripe:	3	0	1	2	6
					(parity)

Parity Functions

Problem C.

Which functions could we use to compute parity?

Remember, disk blocks are finite
(assume modulo arithmetic).

RAID-4: Analysis

0a) What is capacity?

0b) How many disks can fail?

0c) Throughput?

0d) Latency?

Disk0	Disk1	Disk2	Disk3	Disk4
3	0	1	2	6

(parity)

RAID-4: Analysis

0a) What is capacity? **(N-1) * C**

0b) How many disks can fail? **1**

0c) Throughput? **???**

0d) Latency? **D** (i), **2*D** (ii)

RAID-4: Throughput

What is steady-state throughput for
- sequential reads?
- sequential writes?
- random reads?
- random writes?

RAID-4: Throughput

What is steady-state throughput for

- sequential reads? **(N-1) * S**
- sequential writes? **(N-1) * S**
- random reads? **(N-1) * R**
- random writes? **R/2**

RAID-4: Throughput

What is steady-state throughput for
- sequential reads? **(N-1) * S**
- sequential writes? **(N-1) * S**
- random reads? **(N-1) * R**
- random writes? **R/2**

how to avoid parity bottleneck?

RAID-5

RAID-5

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

...

RAID-5: Analysis

0a) What is capacity? **(N-1) * C**

0b) How many disks can fail? **1**

0c) Throughput? **???**

0d) Latency? **D** (i), **2*D** (ii)

RAID-5: Throughput

What is steady-state throughput for
- sequential reads?
- sequential writes?
- random reads?
- random writes?

RAID-5: Throughput

What is steady-state throughput for

- sequential reads? **(N-1) * S**
- sequential writes? **(N-1) * S**
- random reads? **N * R**
- random writes? **N/4 * R**

RAID-5: Throughput

What is steady-state throughput for

- sequential reads? **(N-1) * S**
- sequential writes? **(N-1) * S**
- random reads? **N * R**
- random writes? **N/4 * R**

(N-1) * R
R/2

RAID-4

All RAID

	Reliability	Capacity
RAID-0	0	C*N
RAID-1	1	C*N/2
RAID-4	1	N-1
RAID-5	1	N-1

All RAID

	Read Latency	Write Latency
RAID-0	D	D
RAID-1	D	D
RAID-4	D	2D
RAID-5	D	2D

All RAID

	Read Latency	Write Latency
RAID-0	D	D
RAID-1	D	D
RAID-4	D	2D
RAID-5	D	2D

but RAID-5 can
do more in parallel

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-4	(N-1)*S	(N-1)*S	(N-1)*R	R/2
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-4	(N-1)*S	(N-1)*S	(N-1)*R	R/2
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

RAID-5 is strictly better than RAID-4

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

RAID-0 is always fastest and has best capacity.
(but at cost of reliability)

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

RAID-5 better than RAID-1 for sequential.

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

All RAID

	Seq Read	Seq Write	Rand Read	Rand Write
RAID-0	N * S	N * S	N * R	N * R
RAID-1	N/2 * S	N/2 * S	N * R	N/2 * R
RAID-5	(N-1)*S	(N-1)*S	N * R	N/4 * R

RAID-1 better than RAID-4 for random write.

Summary

Many engineering tradeoffs with RAID.
(capacity, reliability, different types of performance).

H/W RAID controllers can handle crashes easier.

Transparent, deployable solutions are popular.

Announcements

Extra late days!

p3b due Friday.

I/O schedulers tomorrow (discussion).

Watch email for office hours today
(but assume usual if you hear nothing).