ILLINOIS TECH

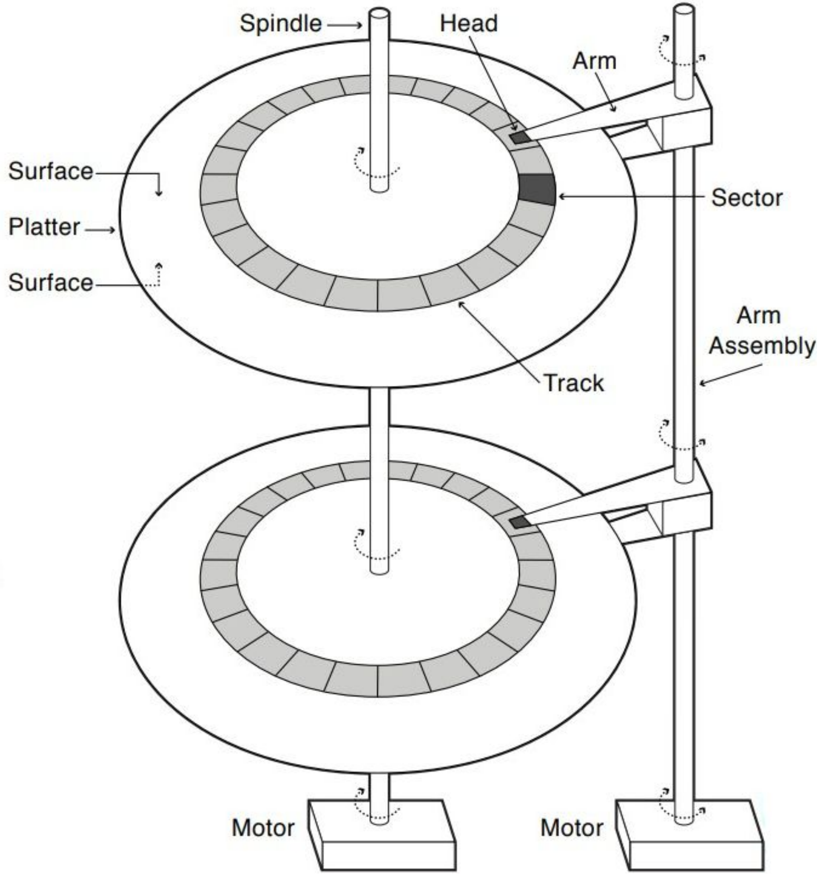
College of Computing

CS 450 Operating Systems
RAID

Yue Duan

Recap: Hard Disks

- Disk components
  - platters
  - surfaces
  - tracks
  - sectors
  - cylinders
  - arm
  - heads
- Operations:
  - seek
  - read
  - write



Recap: Disk Access Time

- Average time to access some target sector approximated by :
  - Taccess = Tavg seek + Tavg rotation + Tavg transfer
- **Seek time** (Tavg seek)
  - Time to position heads over cylinder containing target sector
  - Typical Tavg seek is 3—9 ms
- **Rotational latency** (Tavg rotation)
  - Time waiting for first bit of target sector to pass under r/w head
  - Tavg rotation = 1/2 x 1/RPMs x 60 sec/1 min
- **Transfer time** (Tavg transfer)
  - Time to read the bits in the target sector
  - Tavg transfer = 1/RPM x 1/(avg # sectors/track) x 60 secs/1 min

Recap: Disk Scheduling

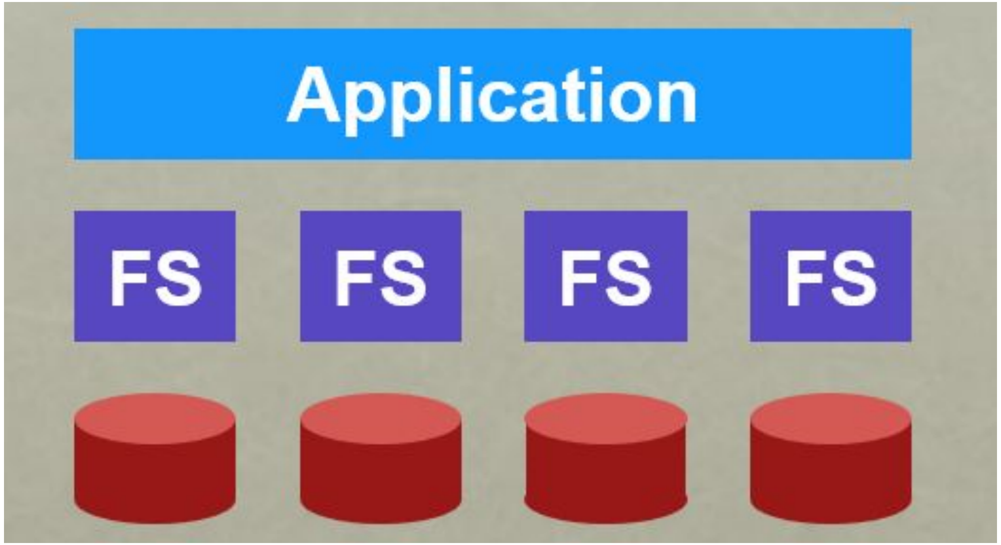
- Seeks are so expensive (milliseconds!)
- OS schedules requests that are queued waiting for the disk
    - **FCFS**
    - **SSTF**
    - **SCAN**
    - **C-SCAN**

Multiple Disks?

- Sometimes we want many disks — why?
  - capacity
  - reliability
  - performance
- Challenge: most file systems work on only one disk

Solution 1: JBOD

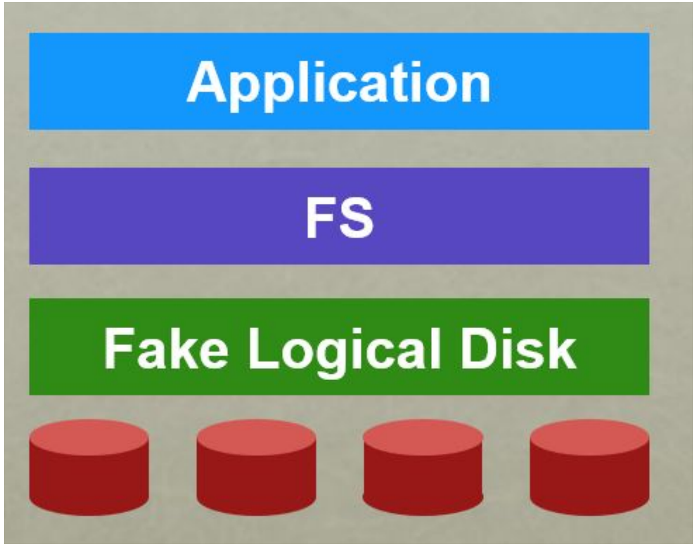
- JBOD: Just a Bunch of Disks
- Application is smart, stores different files on different file systems.



Solution 2: RAID

- RAID: **R**edundant **A**rray of **I**nexpensive **D**isks
- Build a **logical disk** from many physical disks.

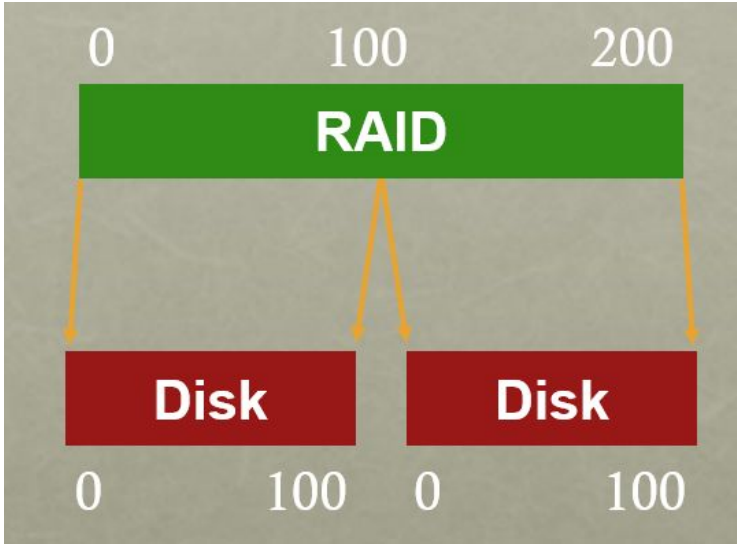
- RAID is
  - transparent
  - deployable



- Logical disk gives
  - capacity
  - performance
  - reliability

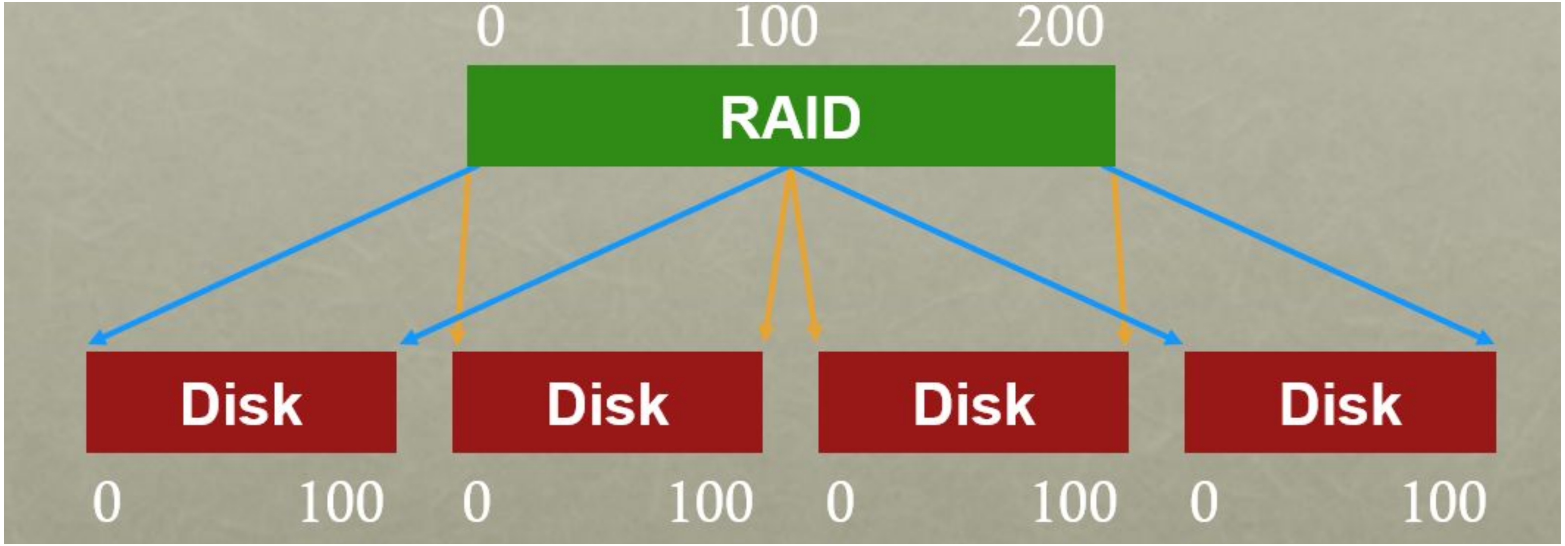
General Strategy: Mapping

- Build a fast, large disk from smaller ones.



General Strategy: Redundancy

- Add even more disks for reliability.



Mapping

- How should we map logical block addresses to physical block addresses?
  - Some similarity to virtual memory
  - 1. Dynamic mapping: use data structure (hash table, tree)
    - page tables
  - 2. Static mapping: use simple math
    - RAID

Redundancy

- Trade-offs to amount of redundancy
- Increase number of copies
  - improves reliability
  - and maybe performance
- Decrease number of copies (deduplication)
  - improves space efficiency

Reasoning about RAID

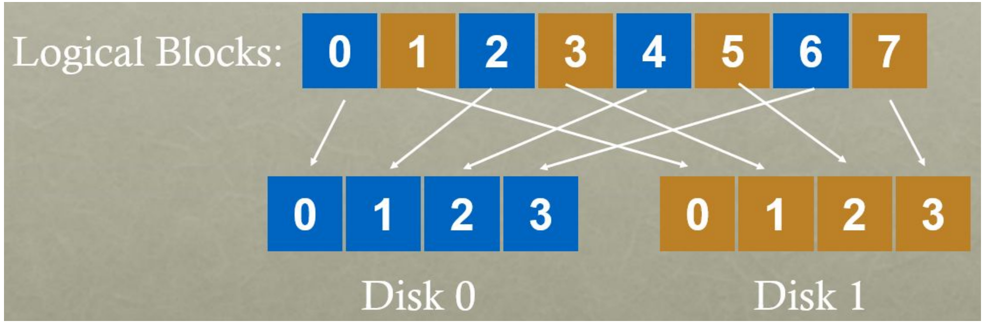
- RAID: system for mapping logical to physical blocks
- Workload: types of reads/writes issued by applications
  - reads, writes
  - sequential vs. random
- Metric: capacity, reliability, performance
- Decisions:
  - Which logical blocks map to which physical blocks?
  - How do we use extra physical blocks (if any)?
  - Different RAID levels make different trade-offs

Metric

- **Capacity**: how much space can apps use?
- **Reliability**: how many disks can we safely lose?
    - assume fail-stop!
        - each disk works or it doesn't
        - system knows when disk fails
- **Performance**: how long does each workload take?
    - single-request latency
        - reveals how much parallelism can exist
    - steady-state throughput
        - the total bandwidth of many concurrent requests

RAID-0: Striping

- Blocks of the array across the disks in a round-robin fashion.
- Optimize for capacity
- No redundancy.
- Maximize parallelism



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

RAID-0: 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

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

Chunk Sizes

- Chunk size = 1 block

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 blocks

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

stripe:

Chunk Size

- a small chunk size
  - decrease read/write time
    - increase the parallelism of reads and writes to a single file
    - many files will get striped across many disks
  - increase seek time
    - access blocks across multiple disks
- a bigger chunk size
  - increase read/write time
    - less parallelism
  - reduce positioning time
    - a single file fits within a chunk and thus is placed on a single disk

RAID-0: Analysis

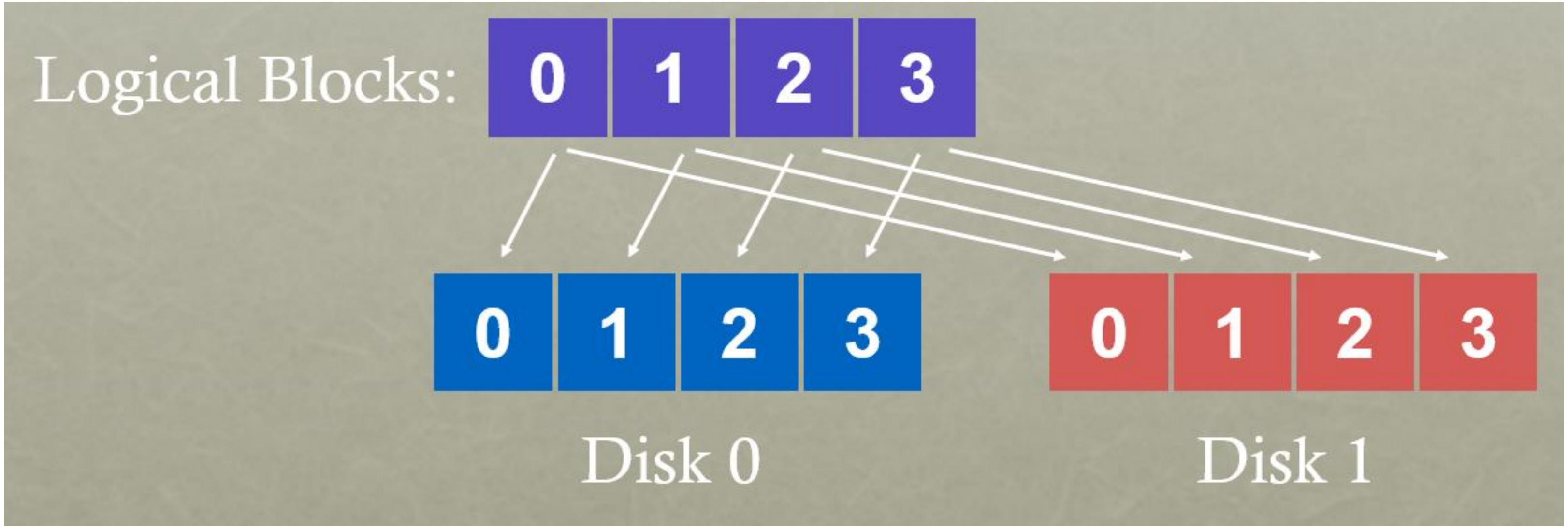
- What is capacity? **N * C**
- How many disks can fail? **0**
- Latency **D**
- Throughput (sequential, random)? **N*S , N*R**

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

Buying more disks improves throughput, but not latency!

RAID-1: Mirroring

- Keep two copies of all data.



RAID-1: Layout

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: 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?
  - Assume disks are fail-stop
    - each disk works or it doesn’t
    - system knows when disk fails
  - Answer: 1

RAID-1: Analysis

- Capacity:
  - N / 2 * C
- How many disks can fail?
  - 1 (or up to N / 2)
- Latency (read, write)
  - ~D

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: Analysis

- What about steady-state throughput?
  - sequential reads?
    - N / 2 * S
  - sequential writes?
    - N / 2 * S
  - random reads?
    - N * R
  - random writes?
    - N / 2 * R

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

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: Strategy

- Use **parity** disk
- In algebra, if an equation has N variables, and N-1 are known, you can often 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.

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

Block0	Block1	Block2	Block3	Parity
00	10	11	10	11
10	01	00	01	10

Example

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)

RAID-4: Analysis

- Capacity:
  - (N - 1) * C
- How many disks can fail?
  - 1
- Latency (read, write)
  - D, 2 * D (read and write parity disk)

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

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
    - (N-1) * S
  - sequential writes
    - (N-1) * S
  - random reads
    - (N-1) * R
  - random writes
    - **R/2** (read and write parity disk)

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 (Random Writes)

- Additive parity
  - write to the block
  - read in all of the other data blocks in the stripe in parallel
  - write the new data and new parity
- Subtractive parity
  - read the old data and old parity
  - compare the old data and the new data
  - $P_{new} = (C_{old} \oplus C_{new}) \oplus P_{old}$

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: Throughput (Random Writes)

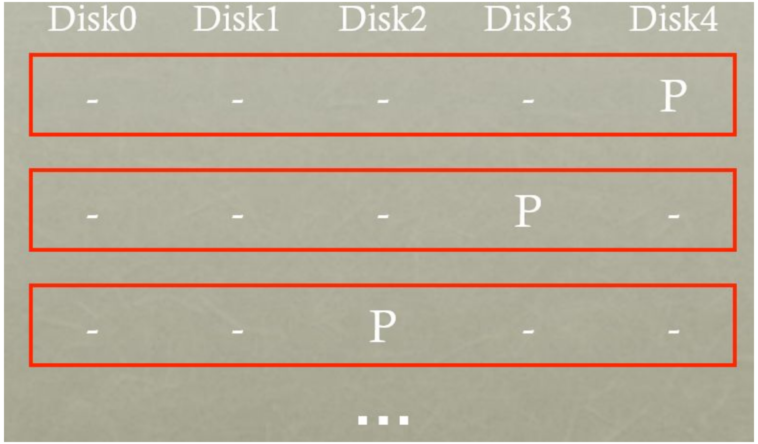
- For each random write
  - perform 4 physical I/Os (two reads and two writes)

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

- Example:
  - write to 4 and 13
  - need to update P1, P3
  - throughput for random writes: R / 2

RAID-5: Strategy

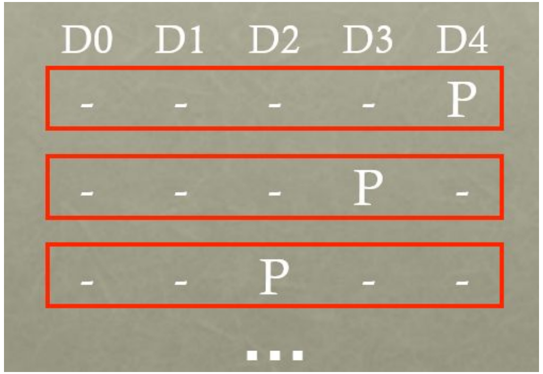
- Rotate parity across different disks



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-5: Analysis

- Capacity:
  - (N - 1) * C
- How many disks can fail?
  - 1
- Latency (read, write)?
  - D, 2*D (read and write parity disk)
- Same as RAID-4



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

- Steady-state throughput for RAID-4:
  - sequential reads: (N-1) * S
  - sequential writes: (N-1) * S
  - random reads: (N-1) * R
  - random writes: R / 2 (read and write parity disk)
- Steady-state throughput for RAID-5:
  - sequential reads: (N-1) * S
  - sequential writes: (N-1) * S
  - random reads: (N) * R
  - random writes: N * R / 4

RAID Level Comparisons

	Reliability	Capacity	Read Latency	Write Latency
RAID-0	0	C*N	D	D
RAID-1	1	C*N/2	D	D
RAID-4	1	(N-1) * C	D	2D
RAID-5	1	(N-1) * C	D	2D

RAID Level Comparisons

	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-0 is always fastest and has best capacity (but at cost of reliability)
- RAID-5 is strictly better than RAID-4
- RAID-5 better than RAID-1 for sequential workloads
- RAID-1 better than RAID-5 for random workloads

THANK YOU!