



RAID

Ohad Rodeh



Introduction

- RAID - Redundant Array of Independent Disks
- A term coined in 1988
- The seminal paper is by: D. Patterson, G. A. Gibson, R. Katz, 1988. **A Case for Redundant Arrays of Inexpensive drives (RAID)**. SIGMOD Conference.
- We will discuss: RAID-{0, 1, 4, 5, 10, 51}



The general idea

- Combine multiple inexpensive disks into one large virtual disk
- Virtual disk appears to be a regular disk to the computer
- The virtual disk will have high performance
- Fault tolerance is supported by using some of the disk-space for an error correcting code, or to store information redundantly

RAID-0

- RAID-0 is striping
 - No redundancy, no fault tolerance
 - High I/O performance, parallel I/O

Disk 1	Disk 2	Disk 3	Disk 4
B_0	B_1	B_2	B_3
B_4	B_5	B_6	B_7

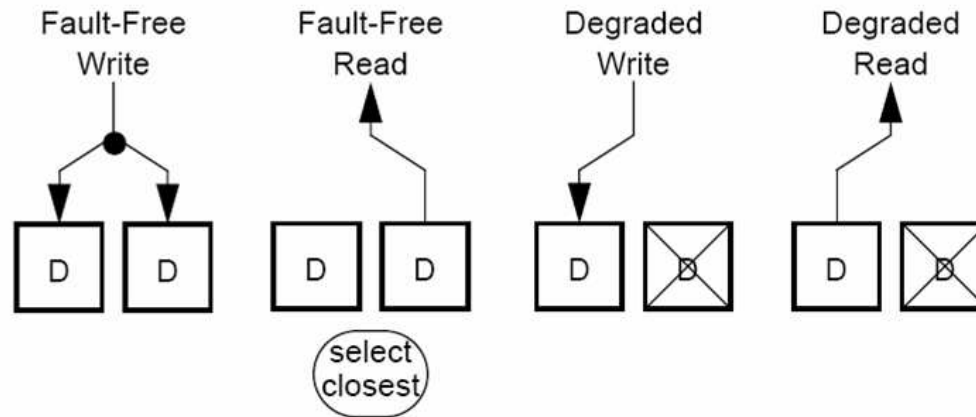
RAID-1

- RAID-1: every block has two copies
 - Tolerates a single disk fault
 - Throughput for writes is $\times \frac{1}{2}$
 - Throughput for reads is $\times 2$
 - Utilization is 50%

Disk 1	Disk 2
B_0	B_0
B_1	B_1
B_2	B_2

Operations in RAID-1

Read and Write Operations in RAID Level 1 (mirroring)



RAID-4

- One disk is used for parity
 - The data is split into equal sized stripes
 - Each stripe is split on $n + 1$ disks

Disk 1	Disk 2	Disk 3	Disk 4	Parity disk
B_0	B_1	B_2	B_3	P_0
B_4	B_5	B_6	B_7	P_1
B_8	B_9	B_{10}	B_{11}	P_2

- A full stripe is a single one row $\{B_0, B_1, B_2, B_3, P_0\}$
- A strip is the part of a stripe on a single disk: B_1



RAID-4 cont.

- Utilization $\frac{n}{n+1}$
- Works well for full-stripe writes, not so well for partial writes
- Works well for reads: can be performed in parallel
- Limitations of RAID-4 parity
 - Can only fix a a single erasure error
 - If a bit is modified, there is no way to figure out which bit this is.



Partial write

- Example: we want to overwrite B_6 with B'_6
- How:
 - Read B_6 and P_1
 - $P'_1 = P_1 \oplus (B_6 \oplus B'_6)$
 - Write B'_6 and P'_1
- Total cost: 2 reads and 2 writes.



Full stripe write

- Full stripe write: overwrite an entire stripe
- Cost: $n + 1$ writes
- The writes can be performed in parallel
- Maximal efficiency for RAID-4



Write atomicity

- When modifying part of a stripe, the entire stripe has to be locked
- There is contention on the parity disk



RAID-4 performance

- Pros:

- Reads work well due to striping
- Full stripe writes work well
- Utilization is $\frac{n}{n+1}$

- Cons:

- Partial writes are bad
- Write performance is limited by the performance of the parity disk

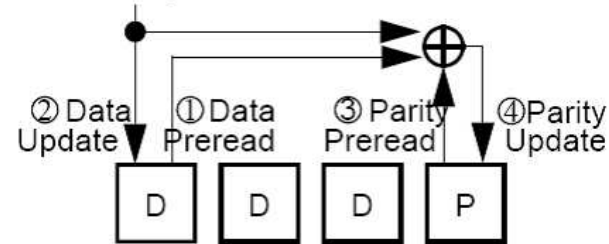
RAID-5

- RAID-5 is the same as RAID-4 except that the parity rotates
- Resolves the bottleneck imposed by the parity disk

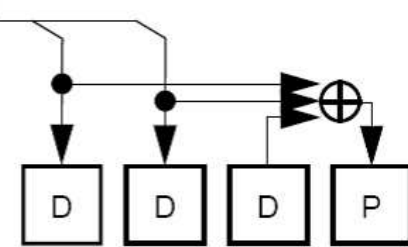
Disk 1	Disk 2	Disk 3	Disk 4	Disk 5
B_0	B_1	B_2	B_3	P_0
B_4	B_5	B_6	P_1	B_7
B_8	B_9	P_2	B_{10}	B_{11}
B_{12}	P_3	B_{13}	B_{14}	B_{15}

Operations in RAID-5

Fault-Free
Read-Modify-Write



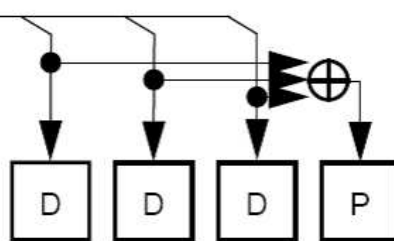
Fault-Free
Reconstruct-Write



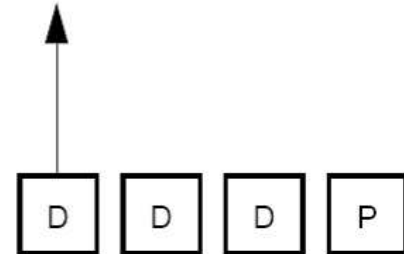
Operations in RAID-5

II

Fault-Free
Large-Write



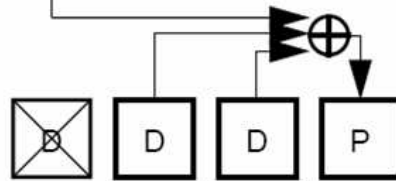
Fault-Free
Read



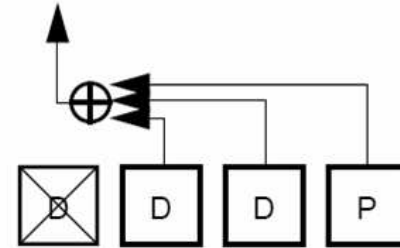
Operations in RAID-5

III

Degraded Write



Degraded Read



RAID-10

- RAID-10 is mirroring + striping

Disk 1	Disk 2	Disk 3	Disk 4
B_0	B_0	B_1	B_1
B_2	B_2	B_3	B_3
B_4	B_4	B_5	B_5

- High performance of RAID-0 and high fault tolerance of RAID-1
- Problems: 50% utilization.



RAID-51

- RAID-51 is RAID-5 + RAID-1
- Two groups of disks, each using RAID-5, and mirroring in-between.
- Utilization: $\frac{1}{2} \times \frac{n}{n+1}$
- Tolerates up to three faults
- Performance: partial writes are inefficient.



Controller behaviour

- Cache behaviour
- Prefetch, the controller tries to guess what will be the next sequence of reads. It is not always successful; it does not have application-level knowledge.
- NVRAM: attempt to store short writes in NVRAM in the hope that they will be made contiguous later on. Then, they can be written to disk using long IOs.



File-systems and RAID

- Which access pattern matches which RAID level?
- Which files do we want to put on which RAID?
- Some examples follow



Regular files

- Use RAID-5
 - Good storage utilization
 - Good for read workloads
 - Can work ok for write if the NVRAM controller cache is effective



Log file

- Use RAID-1.
 - Fault tolerance
 - High write throughput.
 - NVRAM on the controller can convert the short writes to long-sequential IOs to disk.
 - Low storage utilization not a factor. The log is short



Temporary files

- Use RAID-0.
 - No fault tolerance.
 - High throughput.



Write-intensive applications

- Use RAID 10



Performance examples

- The following examples are taken from:
 - DS4500 controller
 - 64KB strip size
 - RAID5 configuration: 8+P, full stripe=512KB, device-size=45BG
 - RAID1 configuration: 2+2, full stripe=128KB, device-size=30BG
 - read/write caching turned off
 - Throughput: measurements taken with 10 threads



Workload

- Random IOs
- 4K or full-stripe aligned
- 10000 rounds



RAID-5 latency

operation	time (msec)	IO/sec
512KB Write	19.070	52.44
512KB Read	12.874	77.68
4KB Write	17.576	56.90
4KB Read	5.297	188.79

RAID-5 throughput

Type	operation	IO/sec
Full-stripe	512KB Write	142.75
	512KB Read	200.87
Non stripe-aligned	512KB Write	63.84
	512KB Read	186.57
Page-aligned	4KB Write	364.66
	4KB Read	1172.61

RAID-1

■ latency

operation	time (msec)	IO/sec
4KB Write	16.814	59.47
4KB Read	3.892	256.95

■ Throughput

operation	IO/sec
4KB Write	640.01
4KB Read	1217.88

RAID-5 with NVRAM, latency

- This is a different disk

operation	IO/sec
512KB Write	57.45
512KB Read	78.43
4KB Write	1841.62
4KB Read	207.68

RAID-5 with NVRAM, throughput

Type	operation	IO/sec
full stripe	512KB Write	55.97
	512KB Read	113.48
Non stripe-aligned	512KB Write	55.83
	512KB Read	108.83
Page-aligned	4KB Write	517.79
	4KB Read	635.00



Summary

- We discussed the basics RAID, simple redundancy schemes
- We covered: RAID-{0,1,4,5,10,51}
- RAID is widely used today
- Understanding RAID is important for the file-system/database/application that is using it.