RAID

Redundant Arrays of Independent Disks

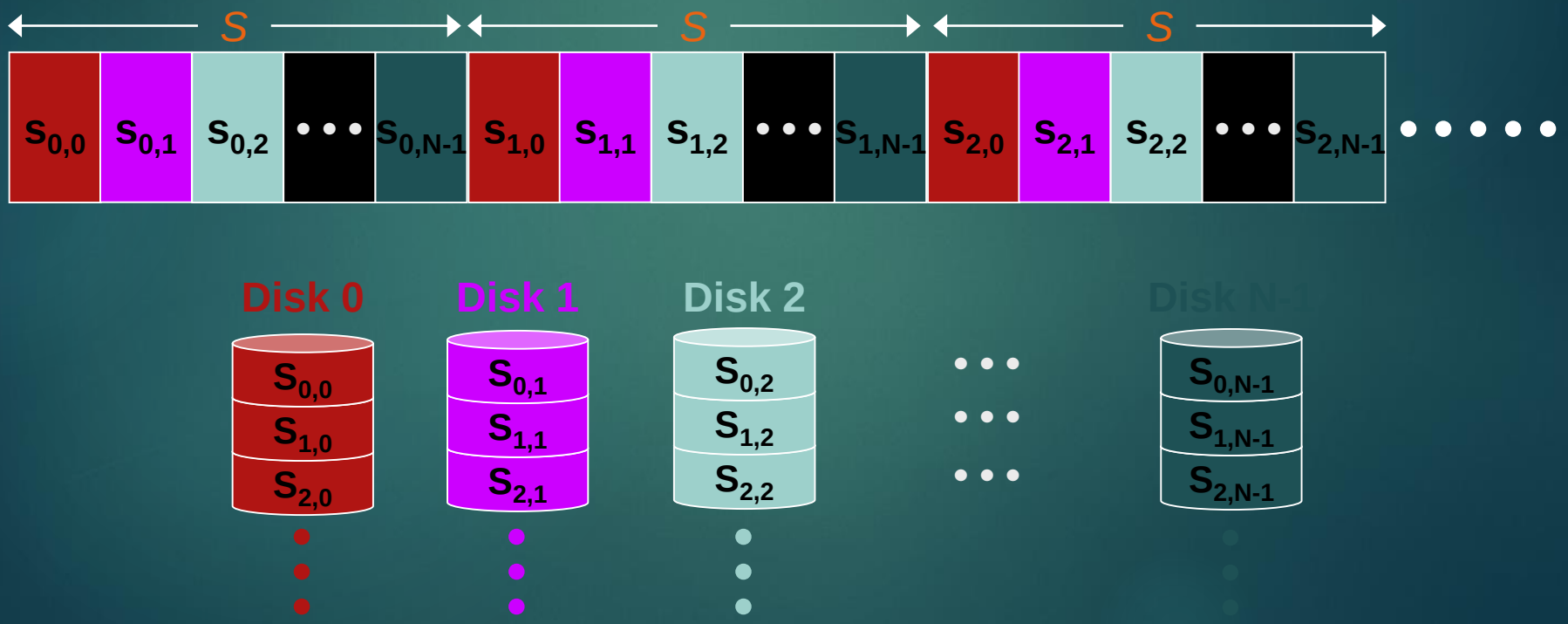
Courtesy of Satya, 15-412 Fall '99
Updated by Gregory Kesden for subsequent classes

Motivation for Disk Arrays

- Typical memory bandwidths ≈ 150 MB/sec
- Typical disk bandwidths ≈ 10 MB/sec
- Result: *I/O-bound applications limited by disk bandwidth*
  - (not just by disk latency!)
- Two common disk bandwidth-limited scenarios
  - Scientific applications: one huge I/O-hungry app
  - Servers: many concurrent requests with modest I/O demands

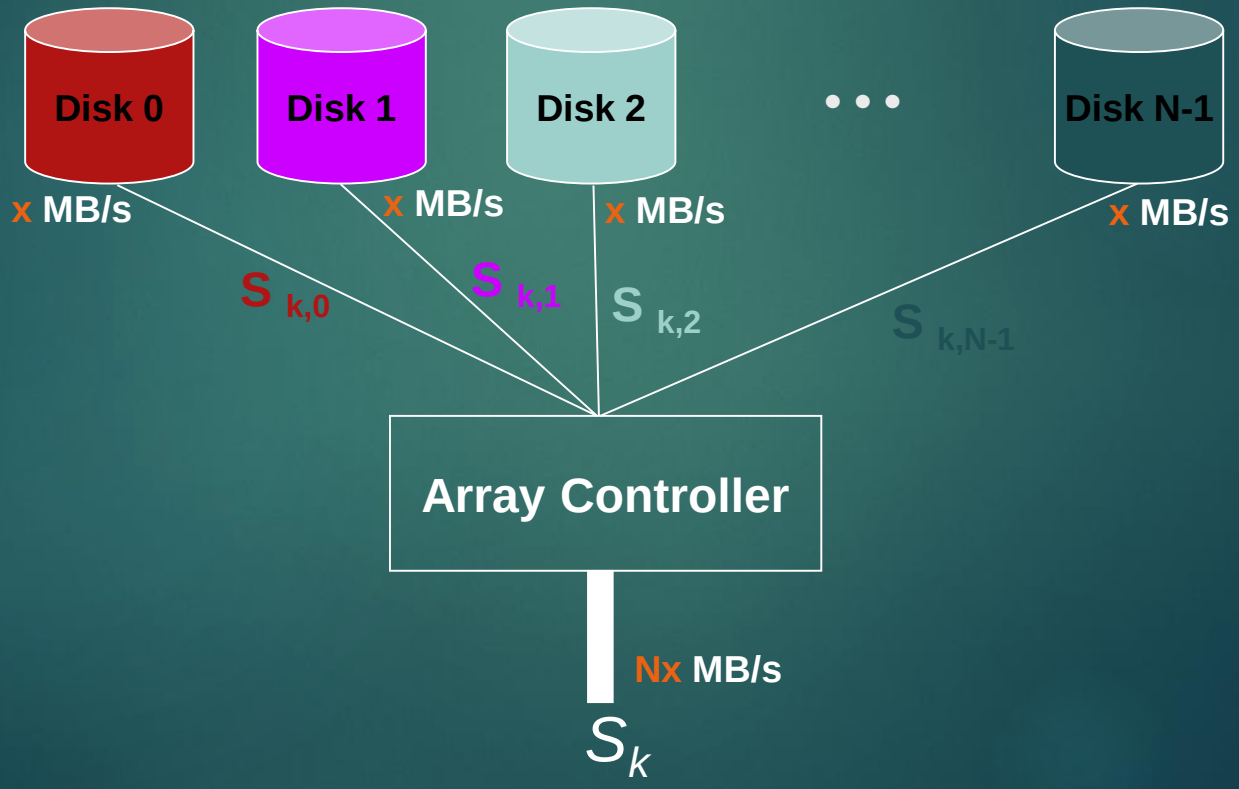
Solution: Exploit Parallelism

- *Stripe* the data across an array of disks
  - many alternative striping strategies possible
- Example: consider a big file striped across N disks
  - *stripe width* is S bytes
  - hence each *stripe unit* is S/N bytes
  - sequential read of S bytes at a time

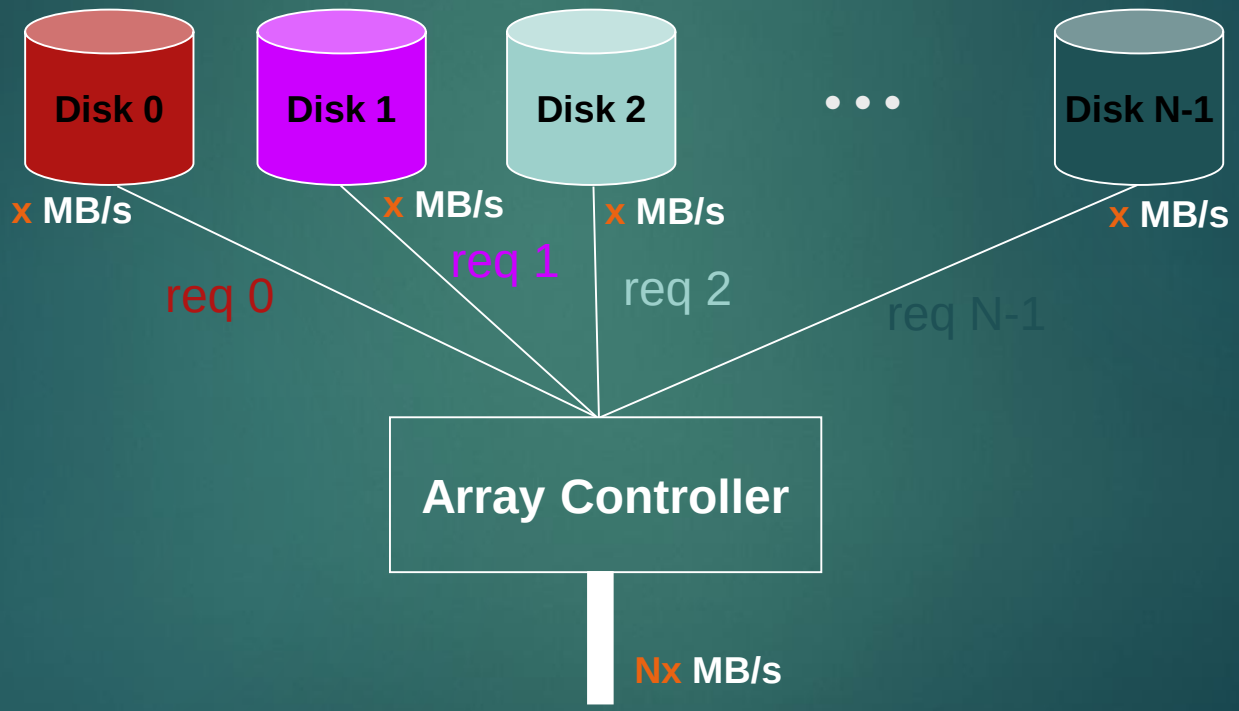


Performance Benefit

- Sequential read or write of large file
  - application (or I/O buffer cache) reads in multiples of S bytes
  - controller performs parallel access of N disks
  - aggregate bandwidth is N times individual disk bandwidth
    - (assumes that disk is the bottleneck)



- *N concurrent small read or write requests*
  - randomly distributed across N drives (we hope!)
  - common in database and Web server environments



N independent requests

Reliability of Disk Arrays

- As number of disks grows, chances of at least one failing increases

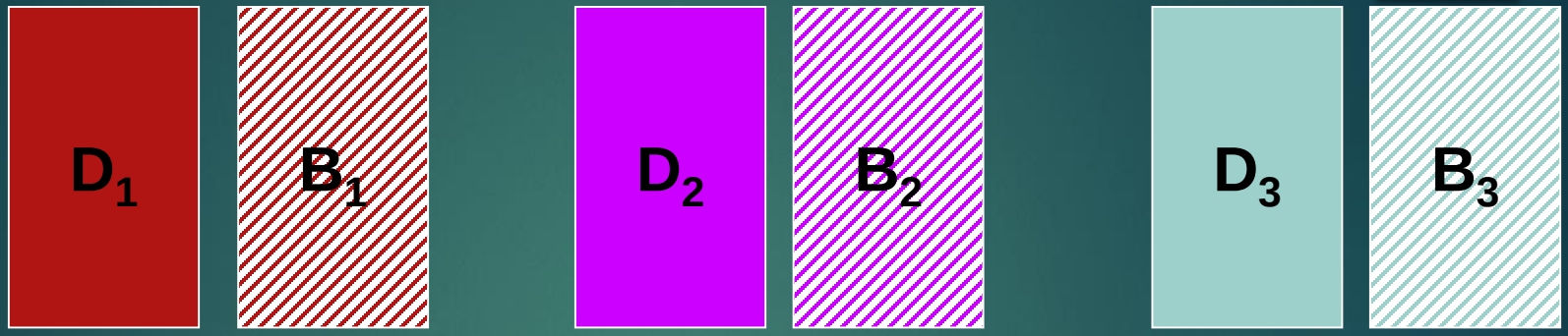
- Reliability of N disks = (reliability of 1 disk) / N
  - suppose each disk has MTTF of 50,000 hours
    - (roughly 6 years before any given disk fails)
  - then some disk in a 70-disk array will fail in (50,000 / 70) hours
    - (roughly once a month!)

- Large arrays without redundancy too unreliable to be useful
  - *"Redundant Arrays of Independent Disks" (RAID)*

RAID Approaches

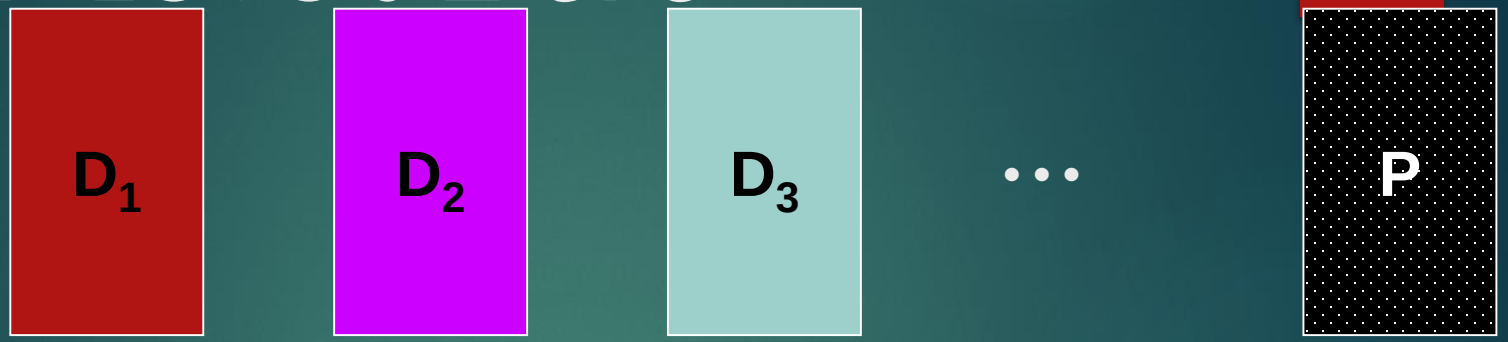
- Many alternative approaches to achieving this redundancy
  - *RAID levels 1 through 5*
  - *hot sparing* allows reconstruction concurrently with accesses
- Key metrics to evaluate alternatives
  - wasted space due to redundancy
  - likelihood of "hot spots" during heavy loads
  - degradation of performance during repair

RAID Level 1



- Also known as *"mirroring"*
- To read a block:
  - read from either data disk or backup
- To write a block:
  - write both data and backup disks
  - failure model determines whether writes can occur in parallel
- Backups can be located far way: safeguard against site failure

RAID Levels 2 & 3



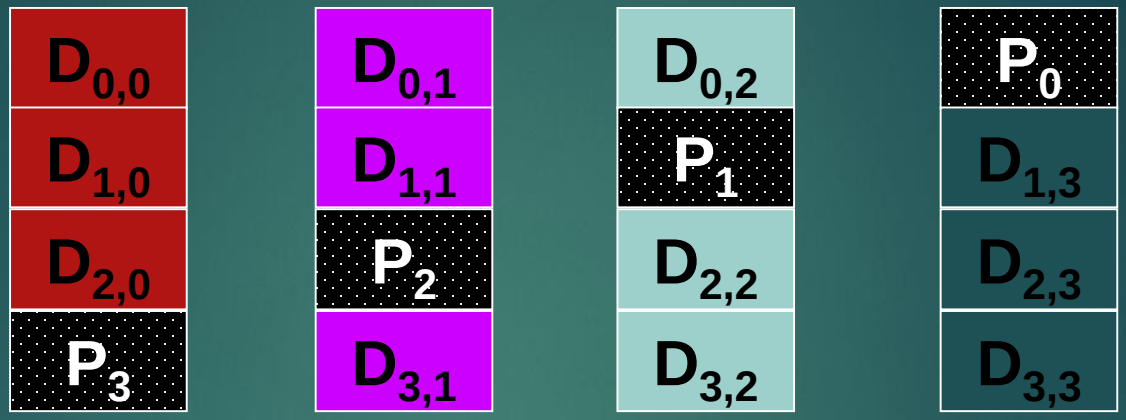
- These are *bit-interleaved* schemes
- In Raid Level 2, P contains memory-style ECC
- In Rail Level 3, P contains simple parity
- Rarely used today

RAID Level 4

$$D_{0,0} \oplus D_{0,1} \oplus D_{0,2} = P_0$$

- *Block-interleaved parity*
- Wasted storage is small: one parity block for N data blocks
- Key problem:
  - parity disk becomes a hot spot
  - write access to parity disk on every write to any block

RAID Level 5

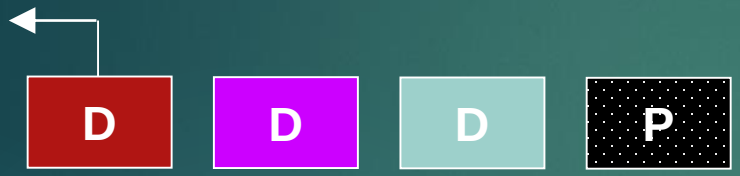


$$D_{0,0} \oplus D_{0,1} \oplus D_{0,2} = P_0$$

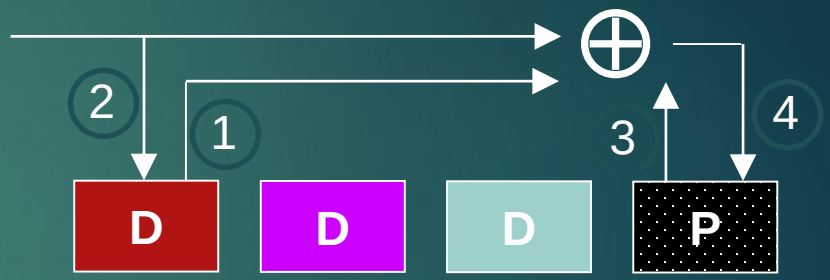
- *Rotated parity*
- Wastage is small: same as in Raid 4
- Parity update traffic is distributed across disks

RAID 5 Actions

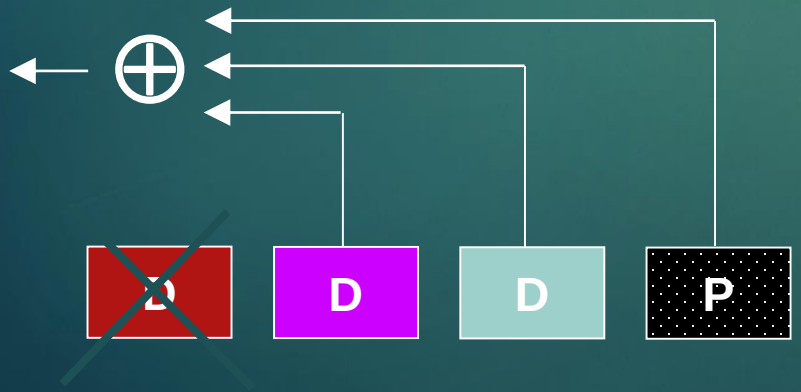
Fault-free Read



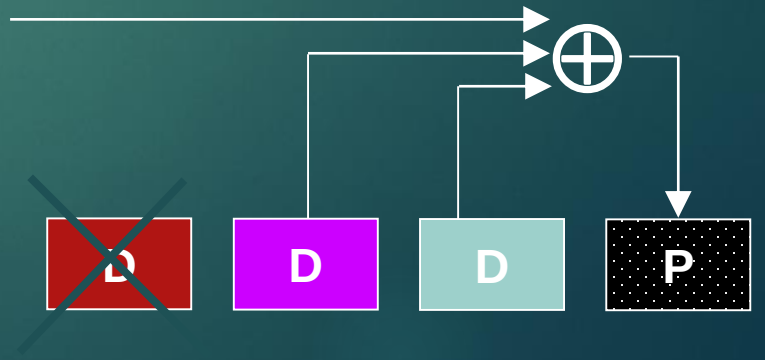
Fault-free Write



Degraded Read



Degraded Write



RAID 6

- Like RAID-5, but has two rotated parity blocks
- Can sustain 2 independent drive failures
- Similarly no read penalty
- More work to do to compute parity
  - For writes
  - Upon failures
- Parity computation isn't simple
  - Uses a Galois Field
  - Much more computational involved
  - Beyond scope of class
  - Key point: Enables reads with 2 failures with only 2 parity blocks