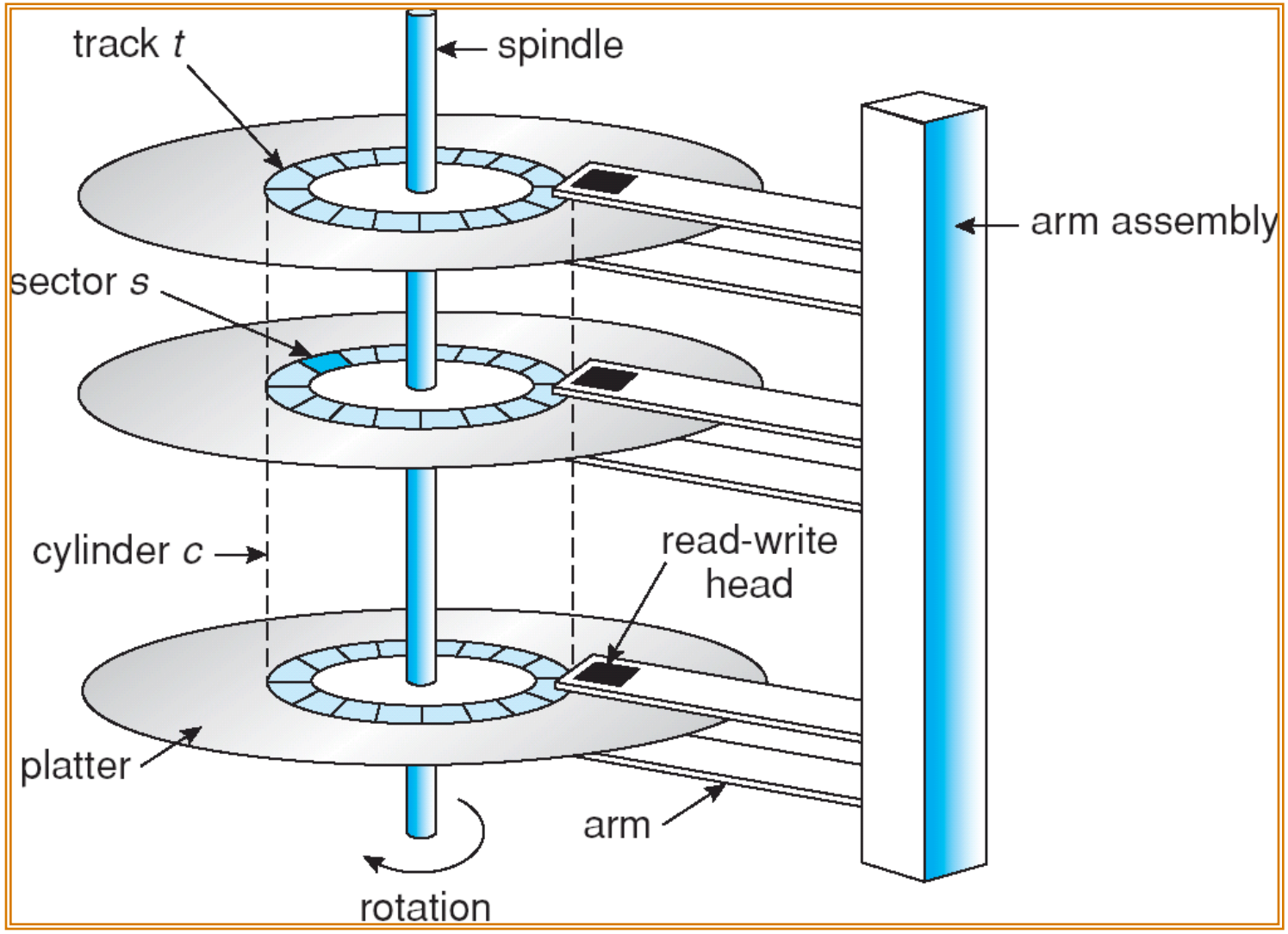
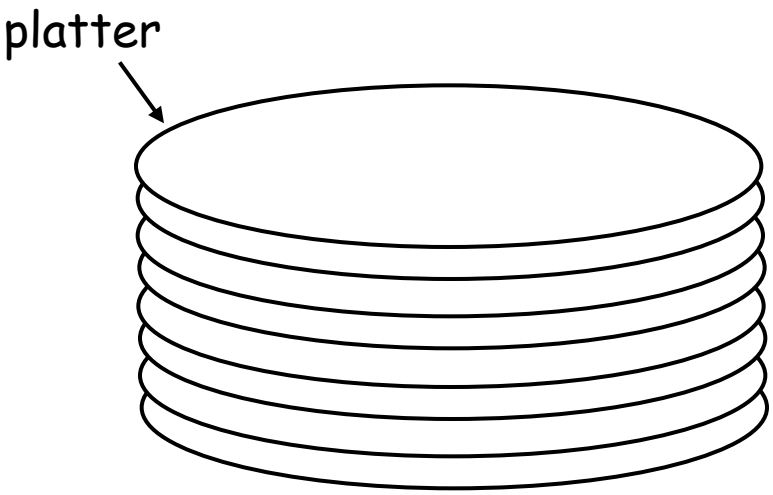
Storage Systems

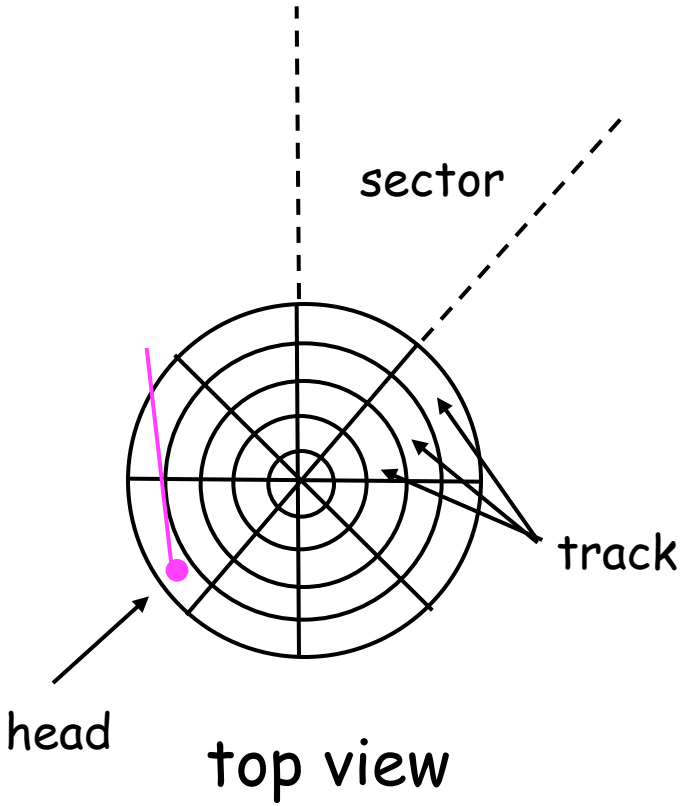
Moving-head Disk Machanism



Disk Structure



cylinder = same track on all platters



Logical addressing: CHS = cylinder, head, sector

Disk Structure

- Disk drives are addressed as large 1-dimensional arrays of *logical blocks*, where the logical block is the smallest unit of transfer.

- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially.
  - Sector 0 is the first sector of the first track on the outermost cylinder.
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.

Disk Access Latency

- Seek time: time to move the disk head to the desired track
  - Initial startup, traverse, settling
- Rotational delay: time to reach desired sector once head is over the desired track
- Transfer rate: rate data read/write to disk
- Some typical parameters:
  - Seek: ~10-15 ms
  - Rotational delay: ~4.15 ms for 7200 rpm
  - Transfer rate: 30 MB/s

Disk Scheduling

- Disks at least four orders of magnitude slower than main memory
- Access time has two major mechanical components
  - *Seek time* move the heads to proper cylinder
  - *Rotational latency* rotate sector under disk head
- Minimize seek time
- Seek time ≈ seek distance
- Disk bandwidth is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer.

Disk Scheduling (Cont.)

- Several algorithms exist to schedule the servicing of disk I/O requests
  - FCFS (fist come first serve)
  - SSTF (shortest seek time first)
  - SCAN
  - C-SCAN
  - C-LOOK

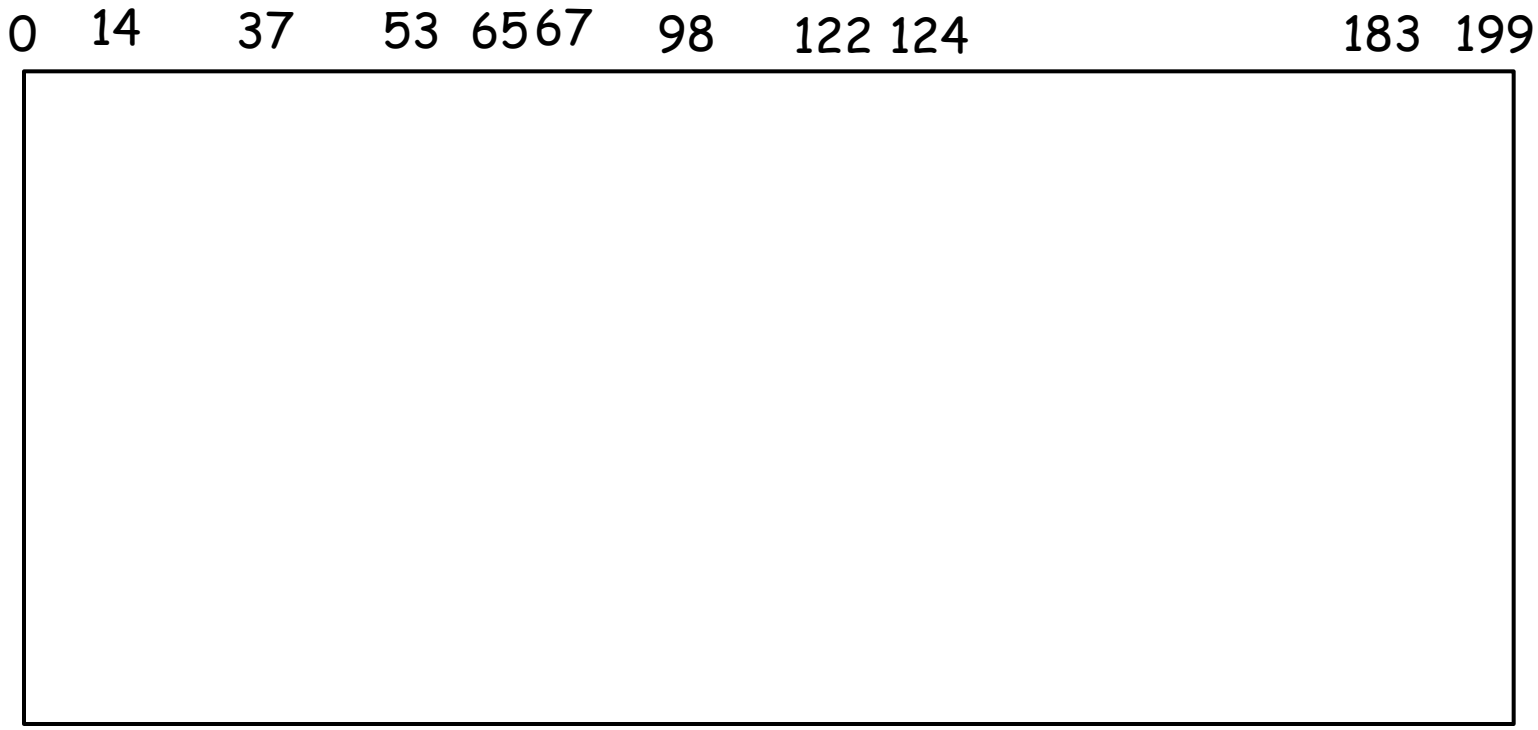
- We illustrate them with a request queue (0-199).

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

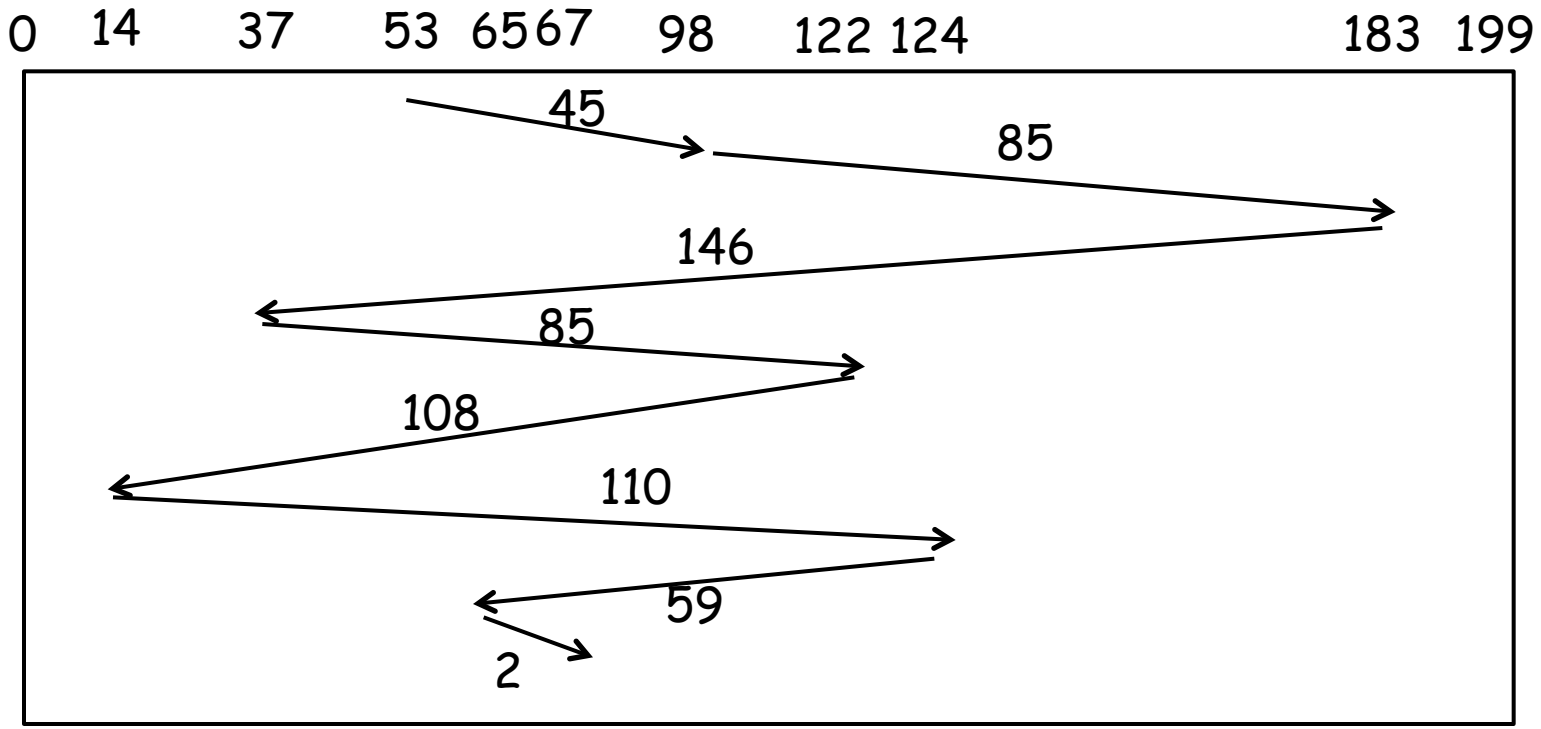
FCFS – First Come First Serve

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



FCFS – First Come First Serve

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



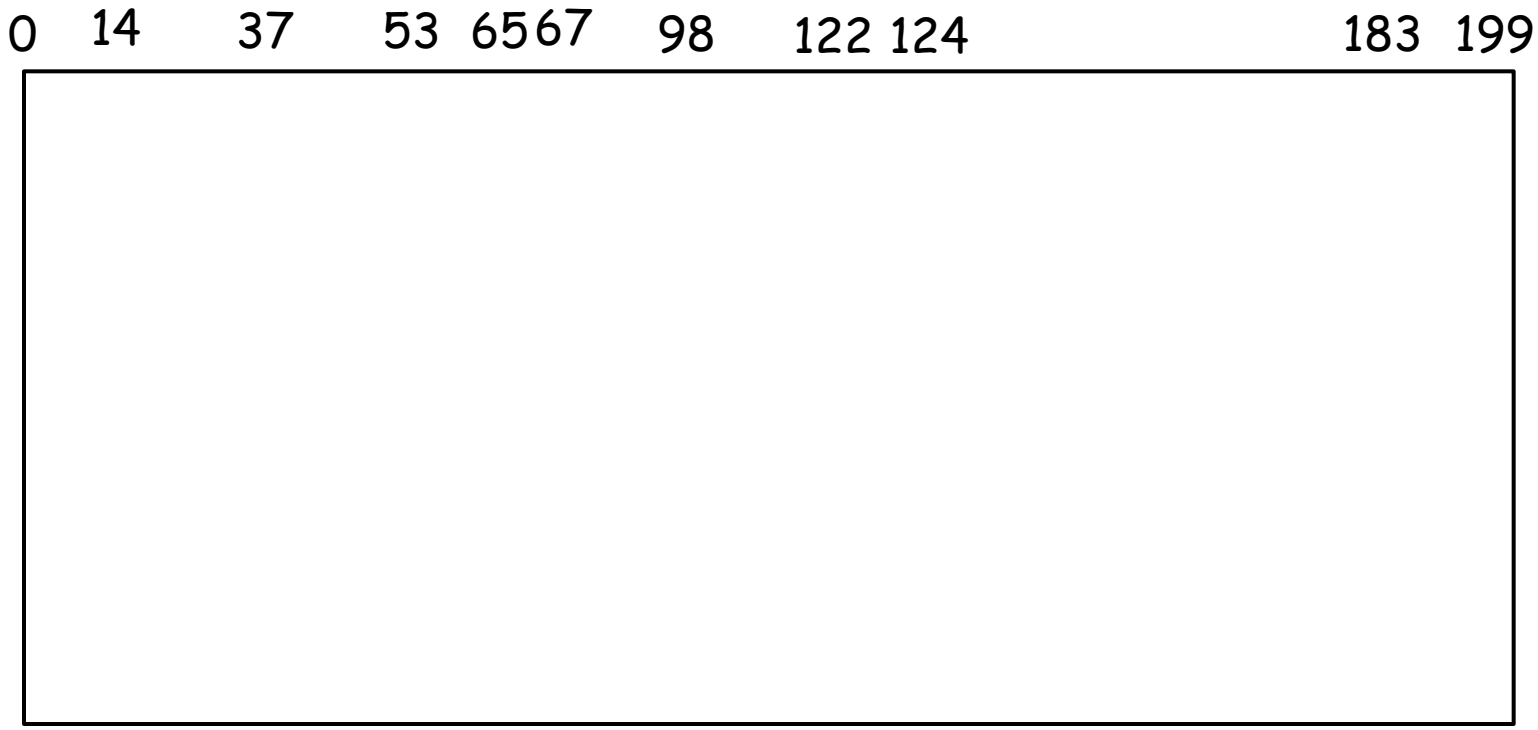
Total seek distance = 640

SSTF (Shortest Seek Time First)

- Selects the request with the minimum seek time from the current head position.
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.

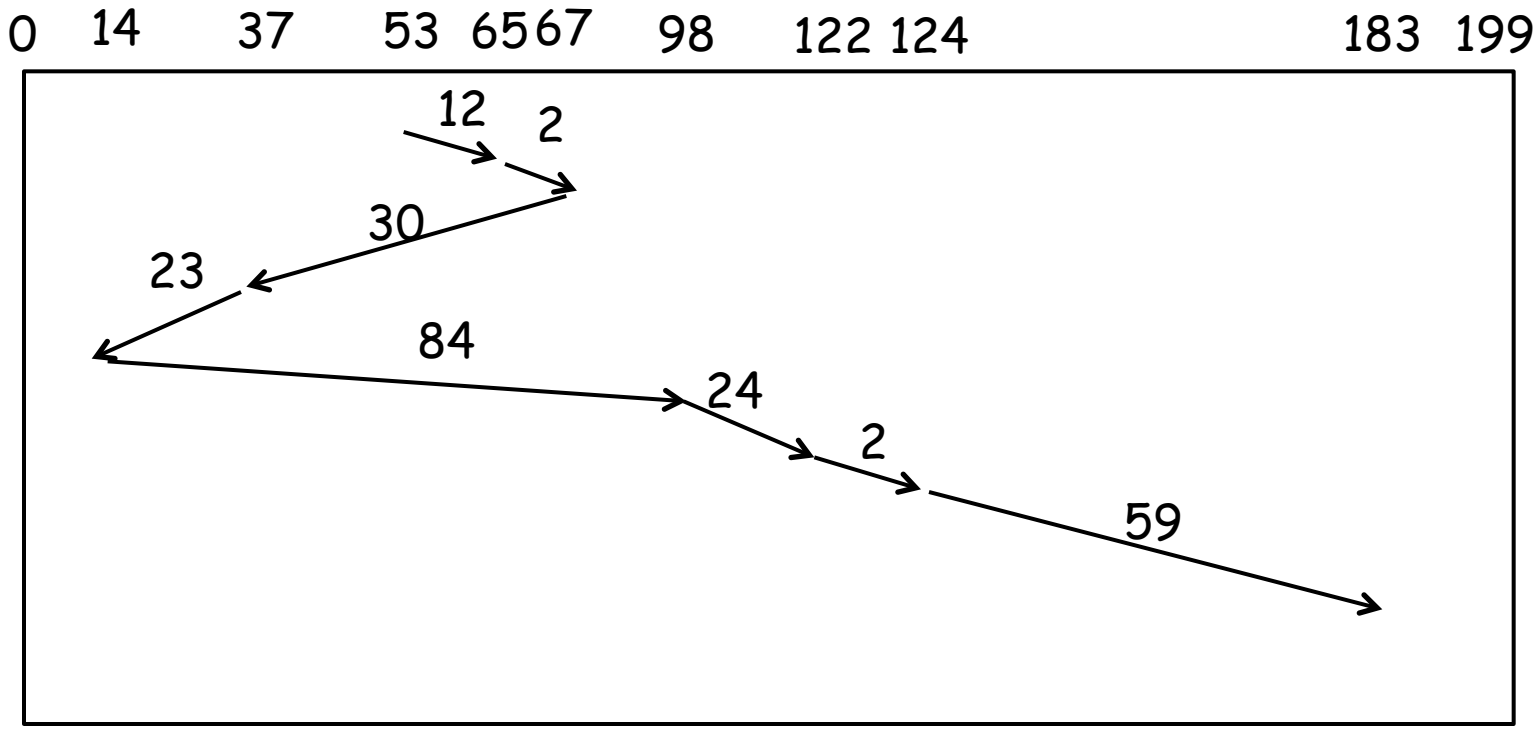
SSTF (Shortest Seek Time First)

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



SSTF (Shortest Seek Time First)

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



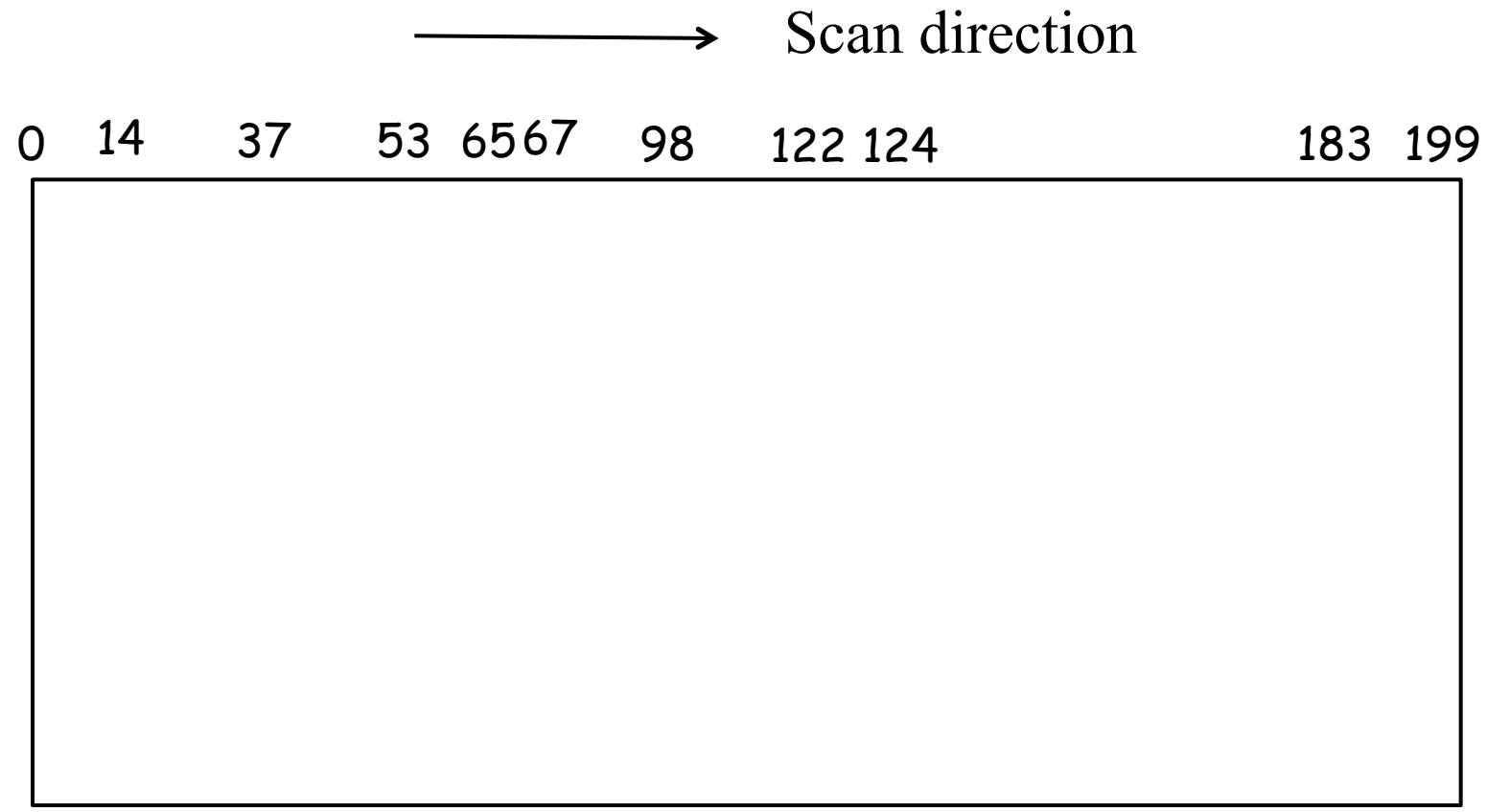
Total seek distance = 236

SCAN

- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes called the *elevator algorithm*.

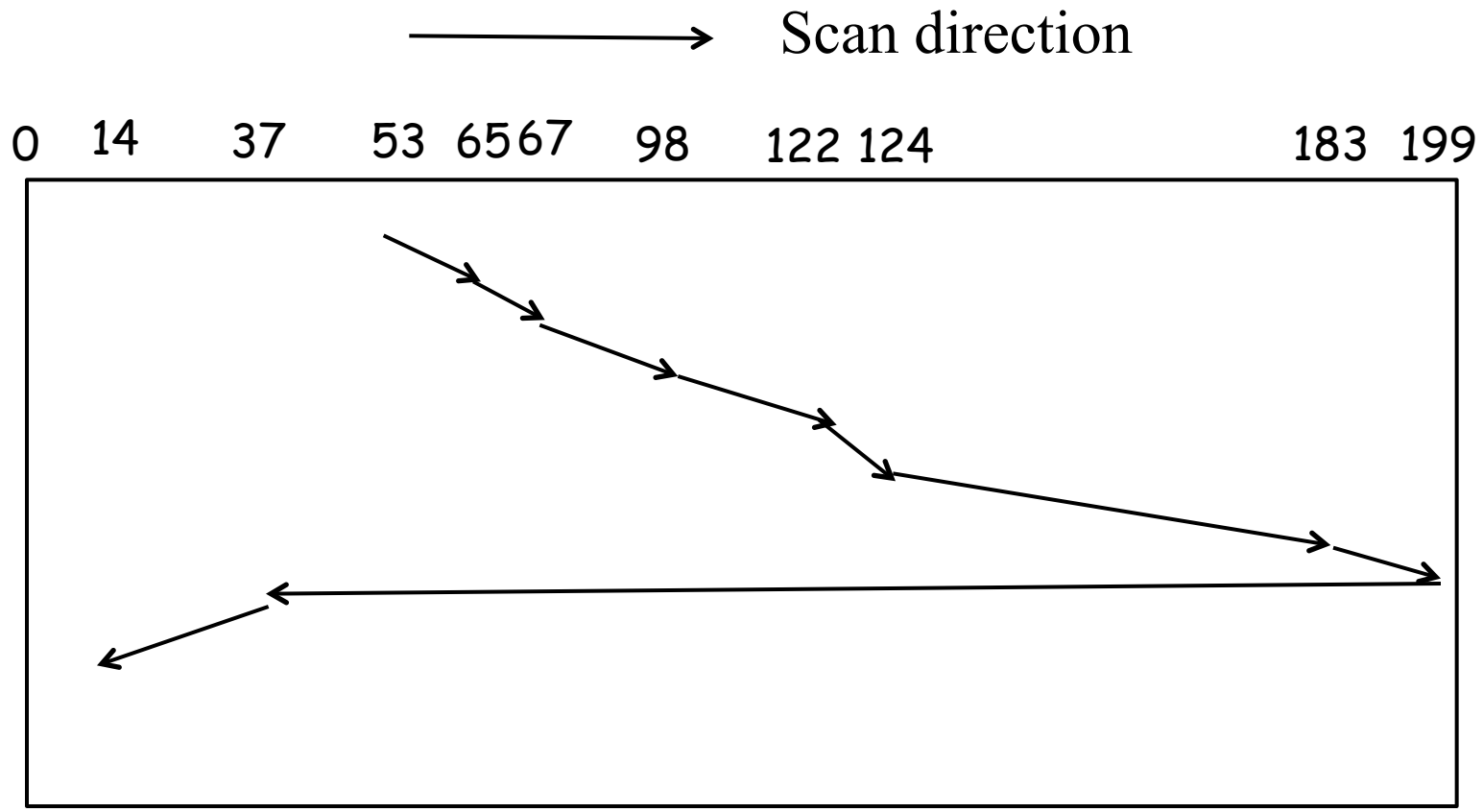
SCAN

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



SCAN

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



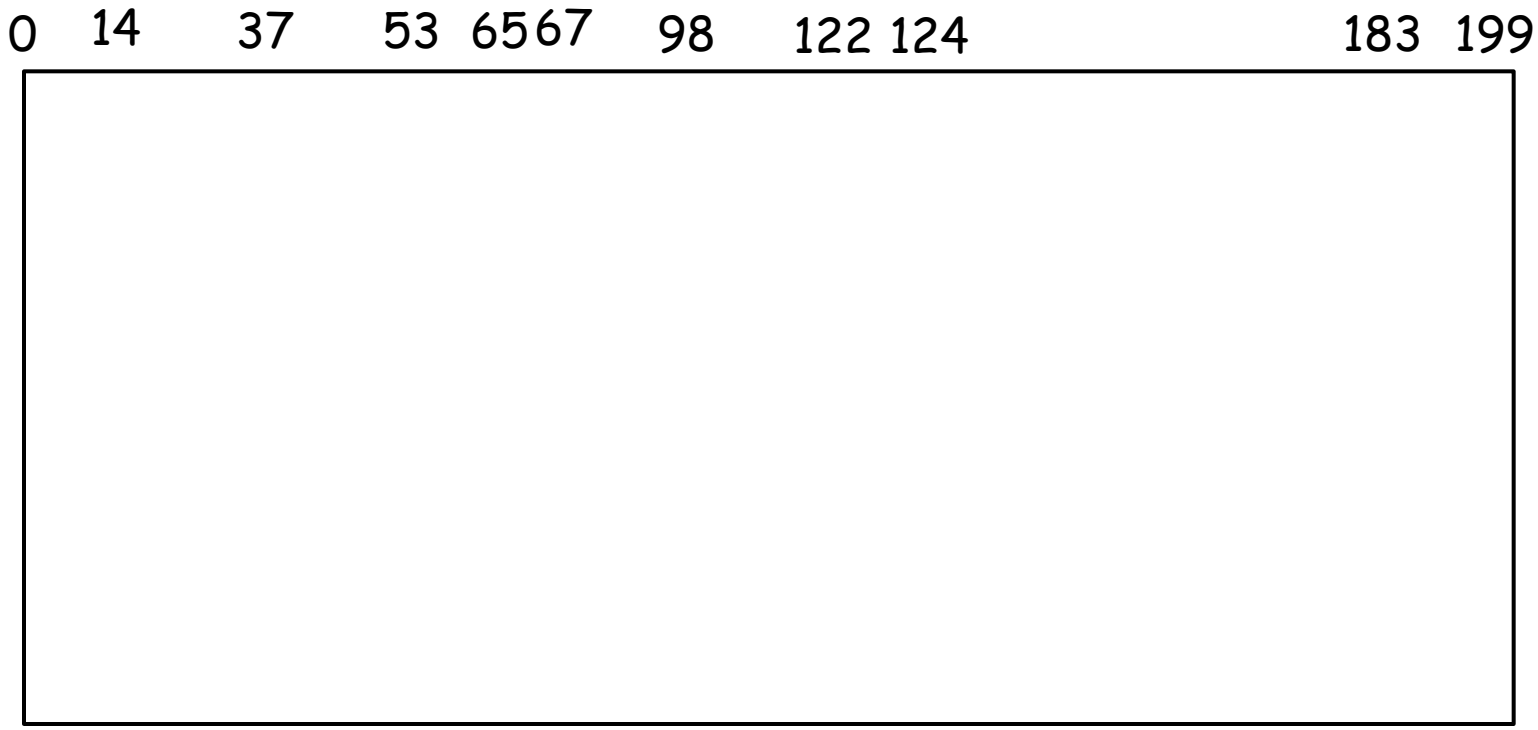
Total seek distance = 331

Circular SCAN (C-SCAN)

- Provides a more uniform wait time than SCAN.
- The head moves from one end of the disk to the other. servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.

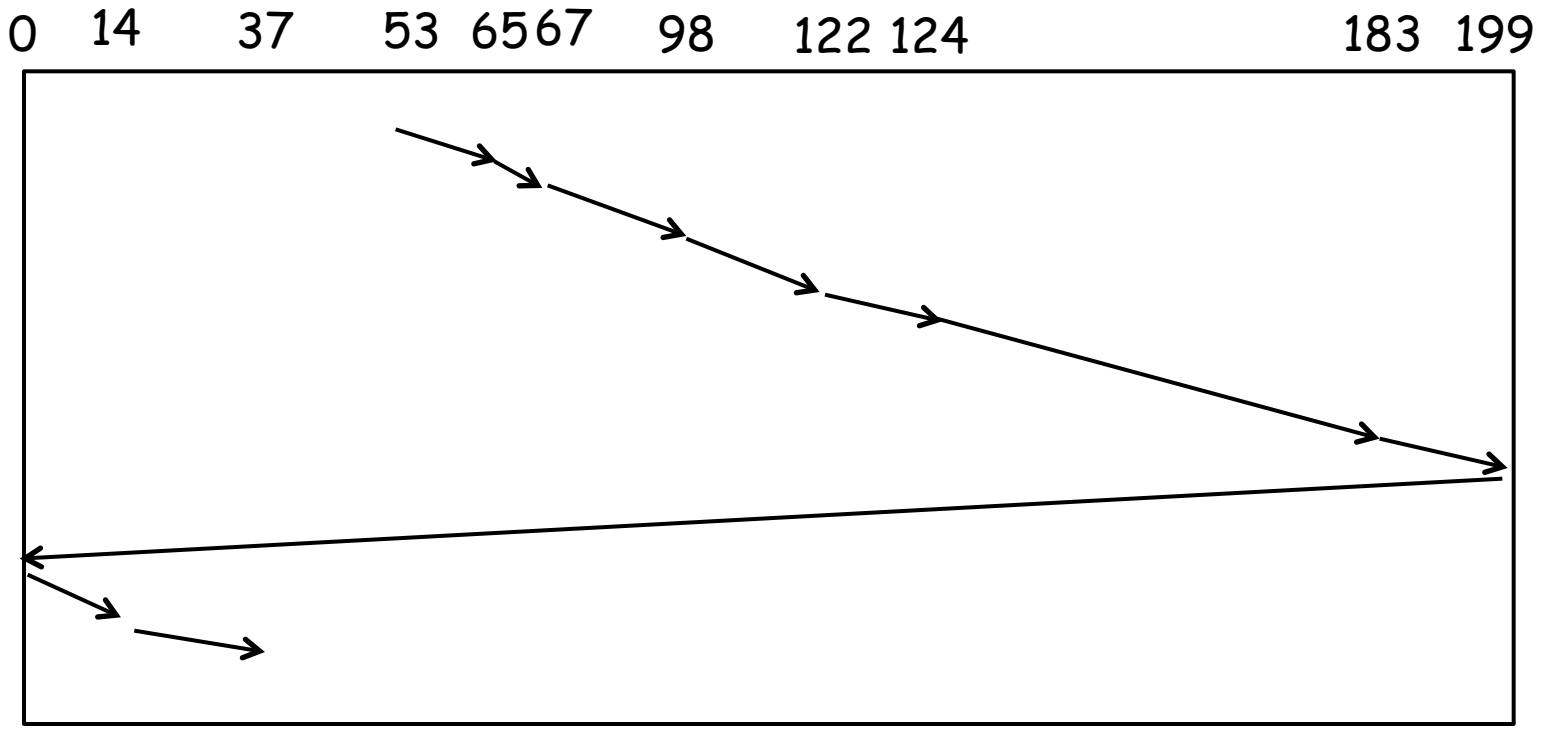
C-SCAN

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



C-SCAN

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



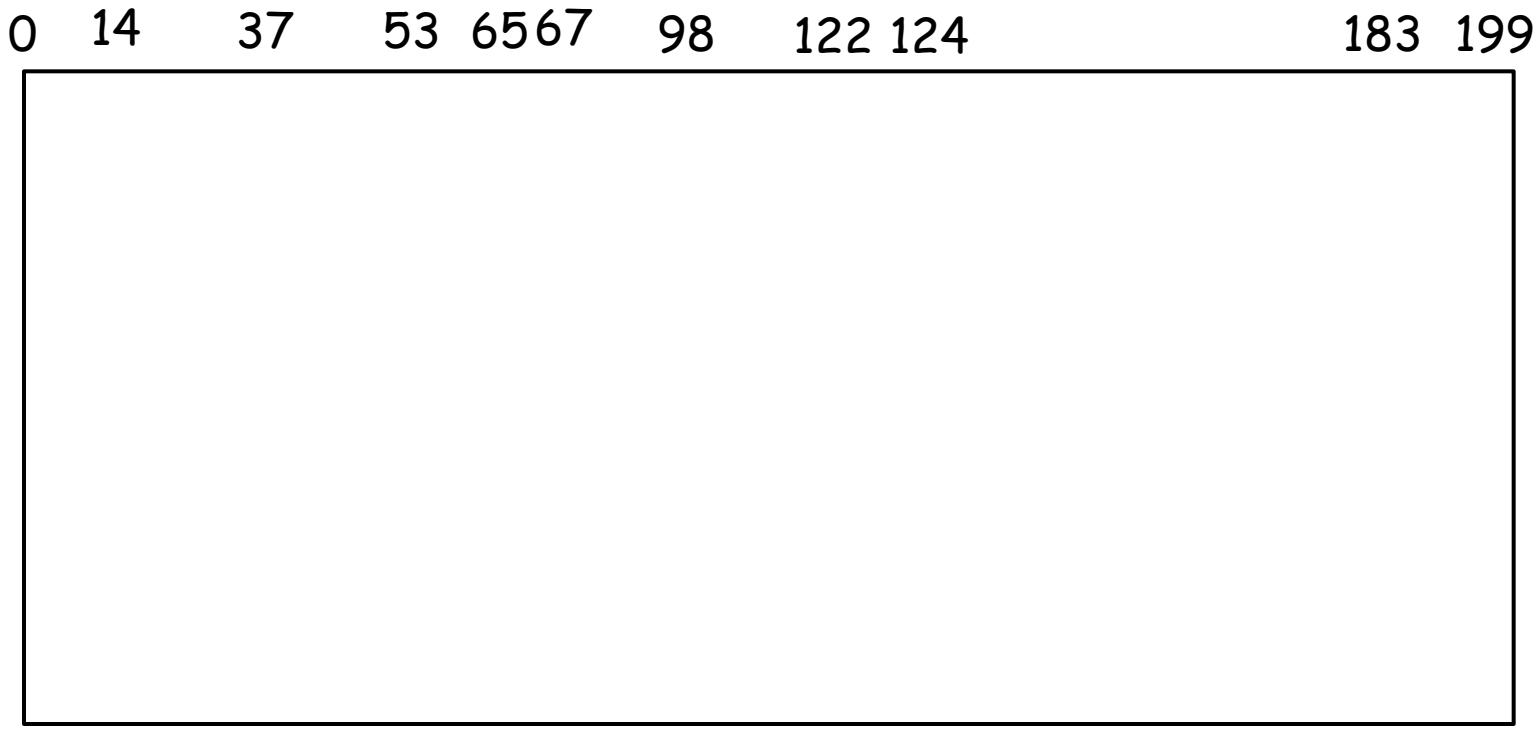
Total seek distance = 383

C-LOOK

- Version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.

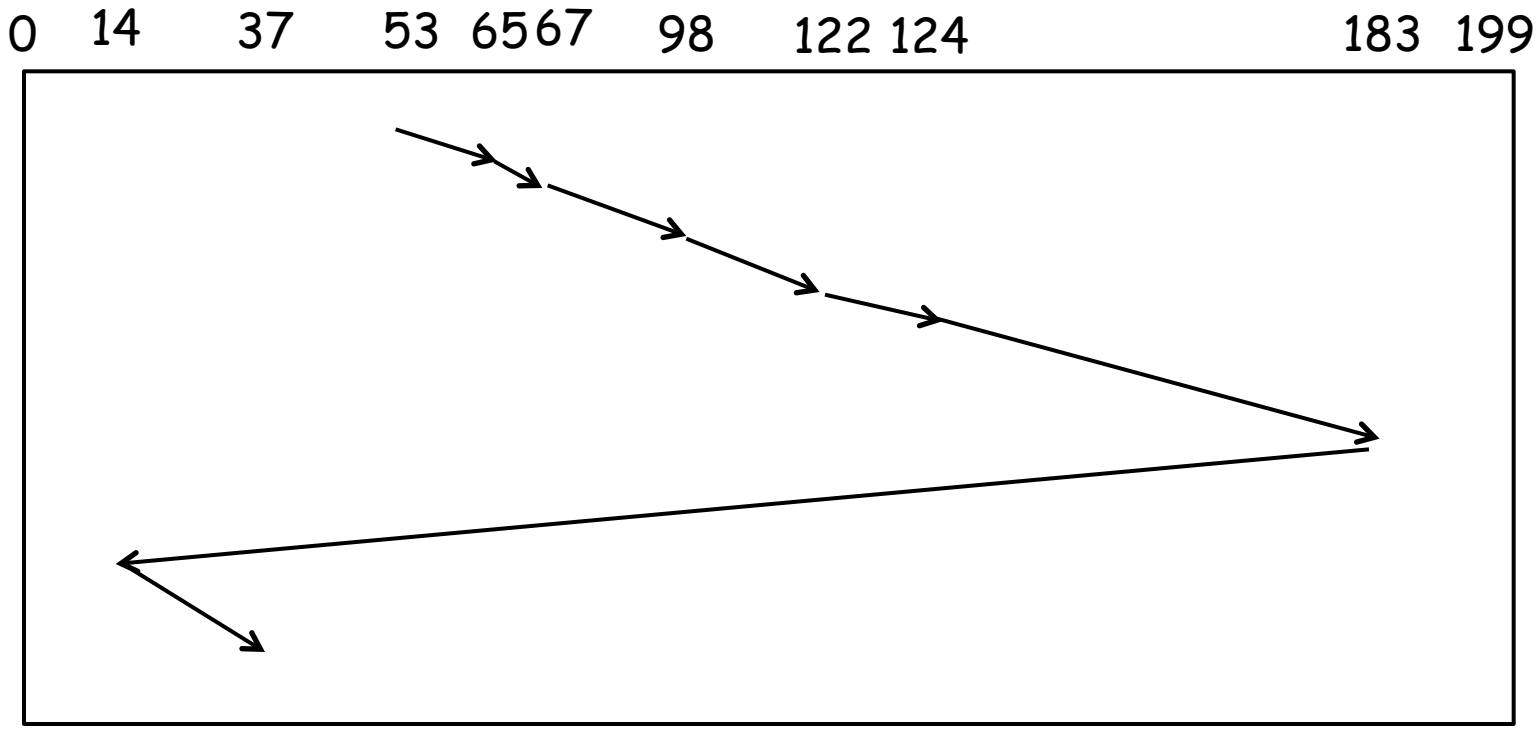
C-LOOK

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



C-LOOK

Request queue = 98, 183, 37, 122, 14, 124, 65, 67, head starts at 53



Total seek distance = 322

Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- Performance depends on the number and types of requests.
- Requests for disk service can be influenced by the file-allocation method.
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
- Either SSTF or C-LOOK is a reasonable choice for the default algorithm.

RAID

- Redundant Array of Inexpensive Disks (RAID)
  - Set of physical disk drives, logically viewed as one (by OS)
  - Replace large-capacity disks with many smaller-capacity
    - Improves I/O performance at lower price
  - Data distributed across physical drives in a way that enables simultaneous access to data from multiple drives
    - Increases throughput
  - Redundant disk capacity is used to compensate for the increase in the probability of failure due to multiple drives
    - Improves availability because no single point of failure
- Six levels of RAID representing different design alternatives

Basis for RAID

- Two RAID aspects taken into consideration:
  - Data striping : leads to enhanced bandwidth
  - Data redundancy : leads to enhanced reliability
    - Mirroring, parity, or other encodings

Data striping

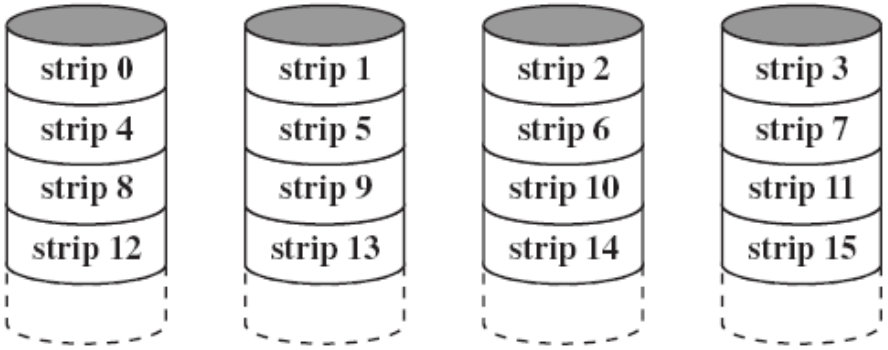
- Data striping:
  - Distributes data transparently over multiple disks
  - Appears as a single fast large disk
  - Allows multiple I/Os to happen in parallel.
- Granularity of data interleaving
  - Fine grained (bit interleaved)
    - Relatively small units; High transfer rates
    - I/O requests access all of disks in the disk array.
    - Only one logical I/O request at a time
    - All disks must waste time positioning for each request: bad!
  - Coarse grained (block-interleaved)
    - Relatively large units
    - Small I/O requests only need a small number of disks
    - Large requests can access all disks in the array

Data redundancy

- Method for computing redundant information
  - Parity, Hamming or Reed-Solomon codes
- Method for distributing redundant information
  - Concentrate on small number of disks vs. distribute uniformly across all disks
  - Uniform distribution avoids hot spots and other load balancing issues.

RAID Level 0

- Does not include redundancy
- Data is striped across the available disks
  - Total storage space across all disks is divided into strips
  - Strips are mapped round-robin to consecutive disks
- Better performance
  - High data transfer capacity due to parallel access and transfer
  - Low I/O response time due to effective request distribution among disks

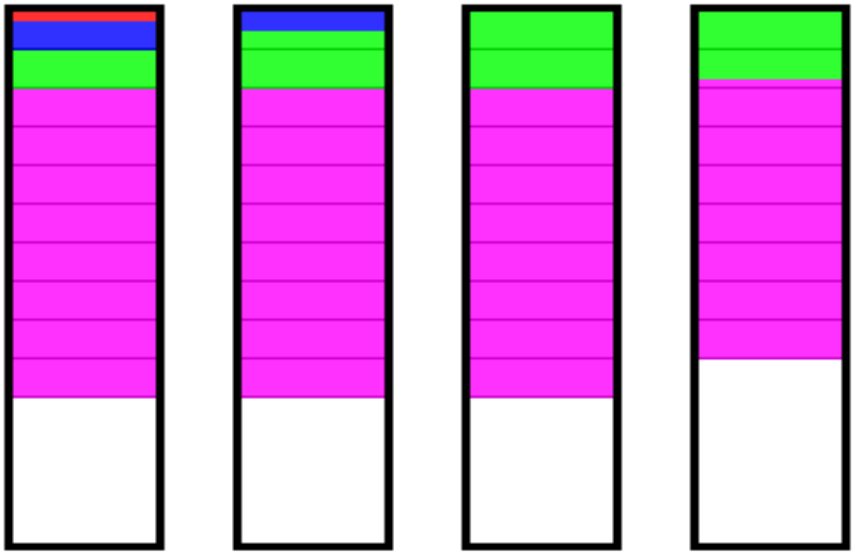


(a) RAID 0 (non-redundant)

RAID Level 0 Practice

- Given four disks, 16KB strip size, assume contiguous allocation, how to place the following four files using RAID0
  - File 1: 4KB
  - File 2: 20KB
  - File 3: 100KB
  - File 4: 500KB

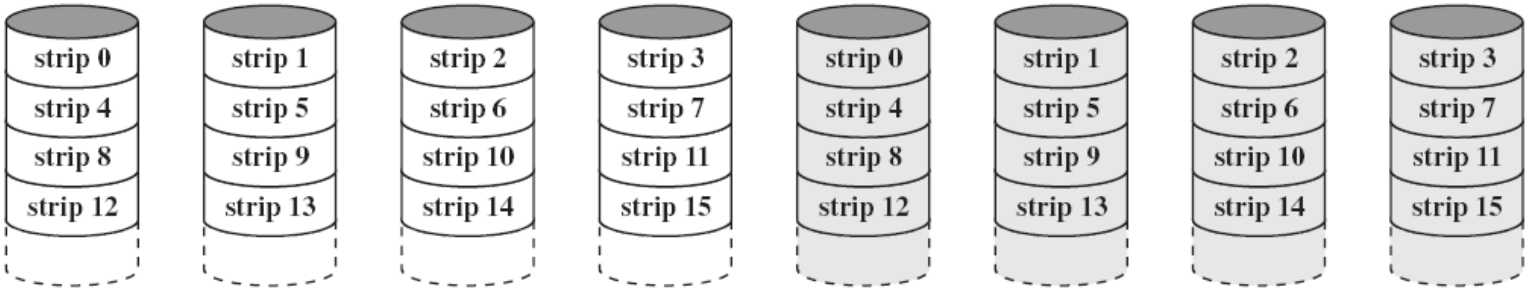
RAID Level 0



Block-interleaved
Four disks, 16KB stripe size.
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Level 1

- Mirroring: Redundancy achieved by duplicating all the data
- Every disk has a mirror disk that stores exactly the same data
  - A read can be serviced by either of the two disks which contains the requested data (improved performance over RAID 0 if reads dominate)
  - A write request must be done on both disks but can be done in parallel
  - Recovery is simple but cost is twice as high

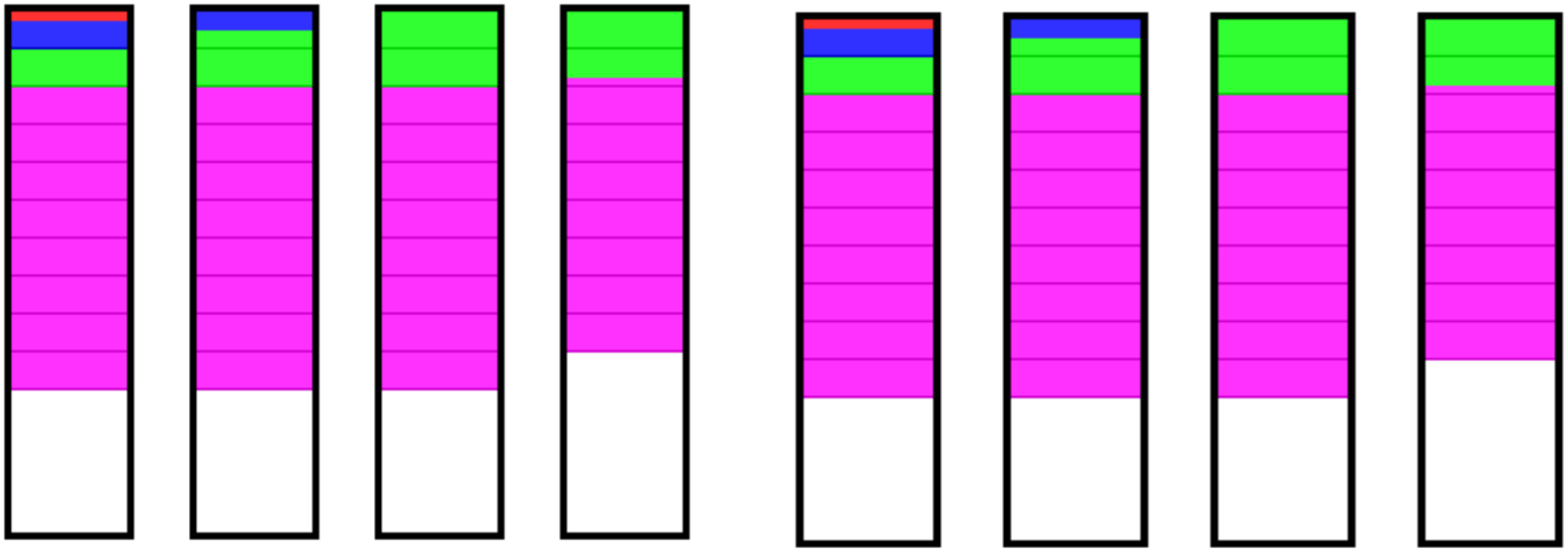


(b) RAID 1 (mirrored)

RAID Level 1 Practice

- Given four disks, 16KB strip size, assume contiguous allocation, how to place the following four files using RAID1
  - File 1: 4KB
  - File 2: 20KB
  - File 3: 100KB
  - File 4: 500KB

RAID Level 1



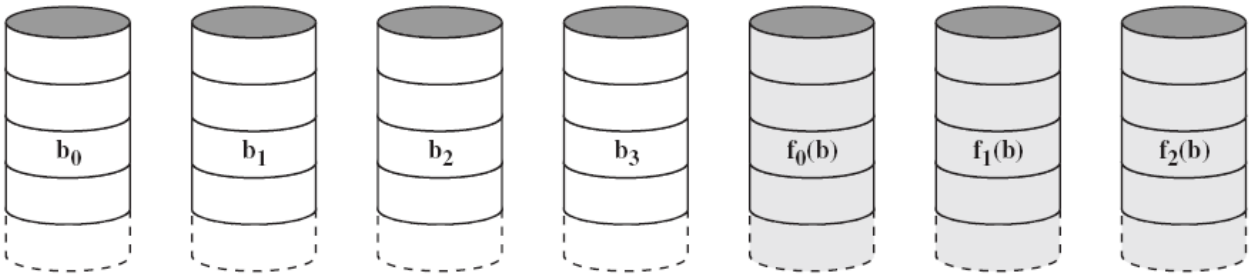
Block-interleaved
eight disks, 16KB strip size.
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Levels 2 and 3: Bit interleaved

- Levels 2 & 3: Parallel access
  - All member disks participate in every I/O request
  - RAID controller assemble data strips and calculate error codes
  - Size of each read/write request = # of disks * strip size
  - High bulk transfer rate, but low transaction rate (one transaction at a time)

RAID Levels 2

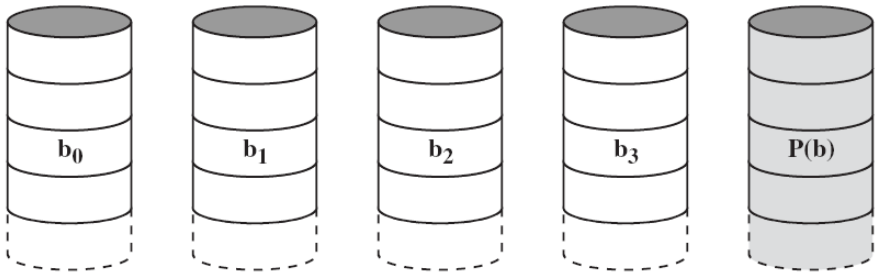
- RAID 2:
  - Error correcting code calculated across corresponding bits on each data disk
  - Hamming code: can correct single-bit errors and detect double-bit errors
  - Less expensive than RAID 1 but still pretty high overhead



(c) RAID 2 (redundancy through Hamming code)

RAID Level 3

- RAID 3: a single redundant disk that keeps parity bits
  - Tolerates only a *single* failure at a time

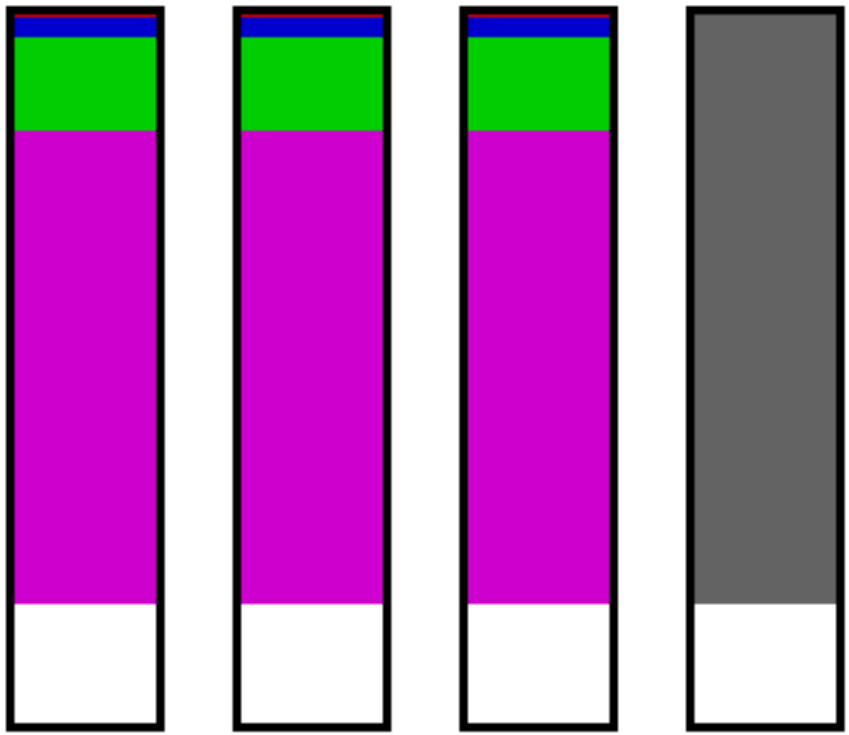


(d) RAID 3 (bit-interleaved parity)

RAID Level 3 Practice

- Given four disks, 16KB strip size, assume contiguous allocation, how to place the following four files using RAID3
  - File 1: 4KB
  - File 2: 20KB
  - File 3: 100KB
  - File 4: 500KB

RAID Level 3



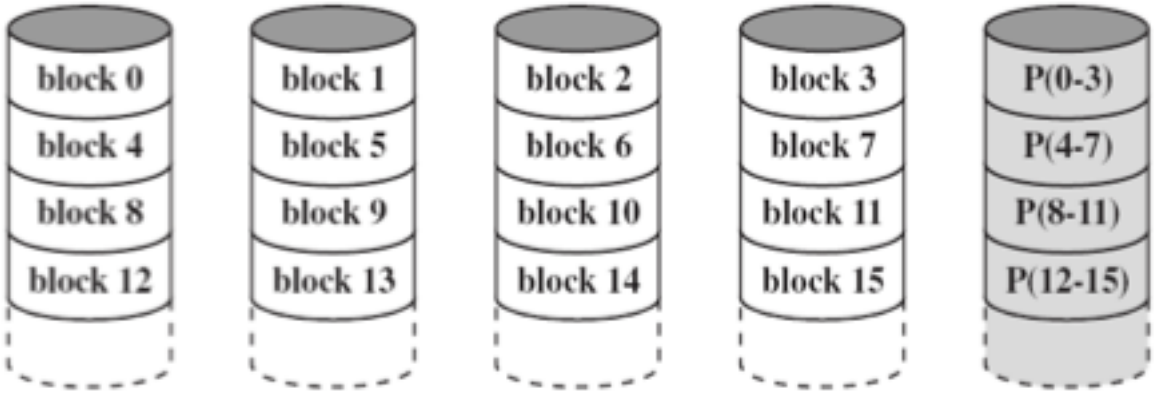
Bit-interleaved
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Levels 4 – 6

- RAID 4 – 6: Independent Access
  - Block interleaved
  - Make use of an independent access technique
  - In an independent access array, each member disk operates independently
  - Multiple I/O requests can be served in parallel
  - Better I/O request rates

RAID Level 4

- RAID 4
  - Large strips with a parity strip like RAID 3
  - Write access → small write = 2 reads + 2 writes (data and parity blocks)

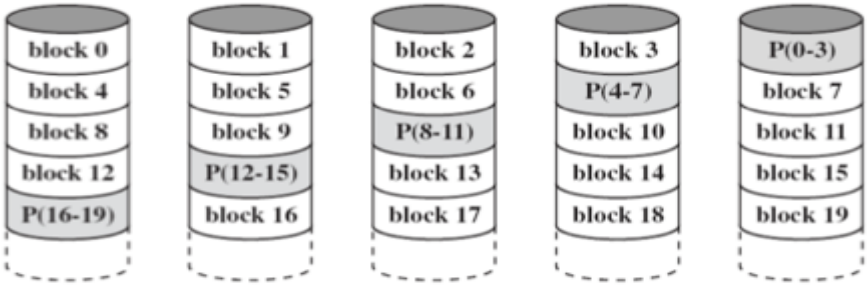


(e) RAID 4 (block-level parity)

Parity disk can become bottleneck

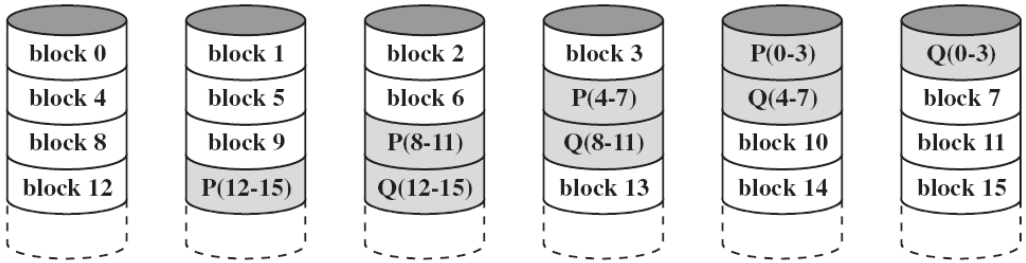
RAID Levels 5 – 6

- RAID 5: parity strips are distributed across all disks



(f) RAID 5 (block-level distributed parity)

- RAID 6: Dual parity strips

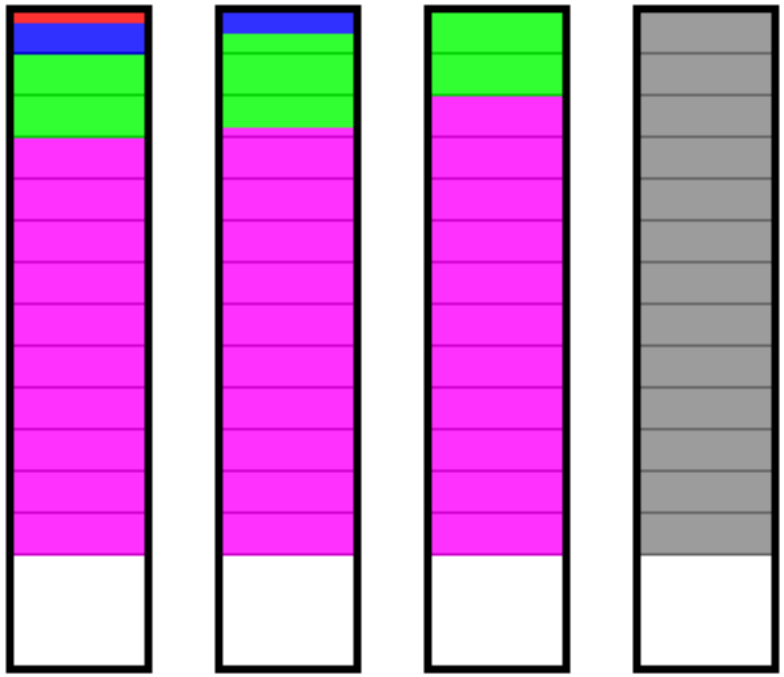


(g) RAID 6 (dual redundancy)

RAID Level 4&5&6 Practice

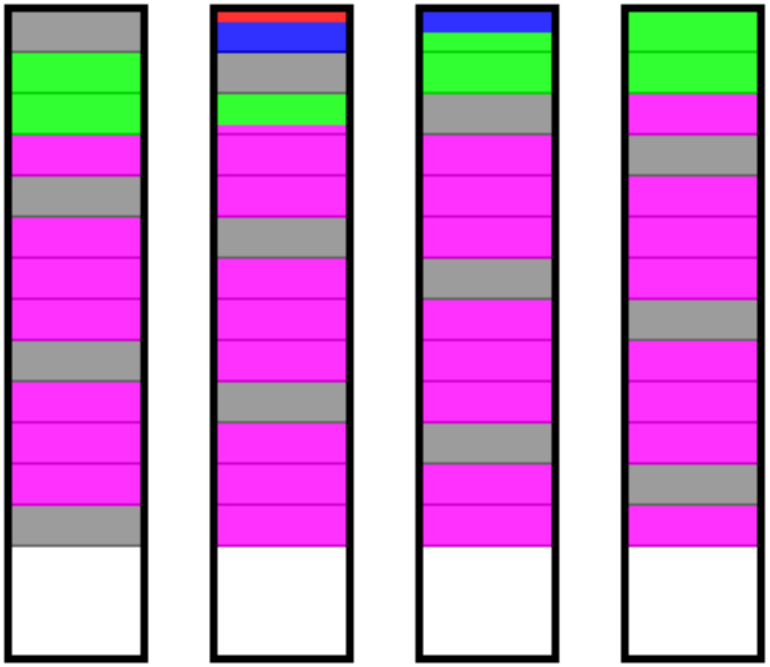
- Given four disks, 16KB strip size, assume contiguous allocation, how to place the following four files using RAID4&5&6
  - File 1: 4KB
  - File 2: 20KB
  - File 3: 100KB
  - File 4: 500KB

Raid Level 4



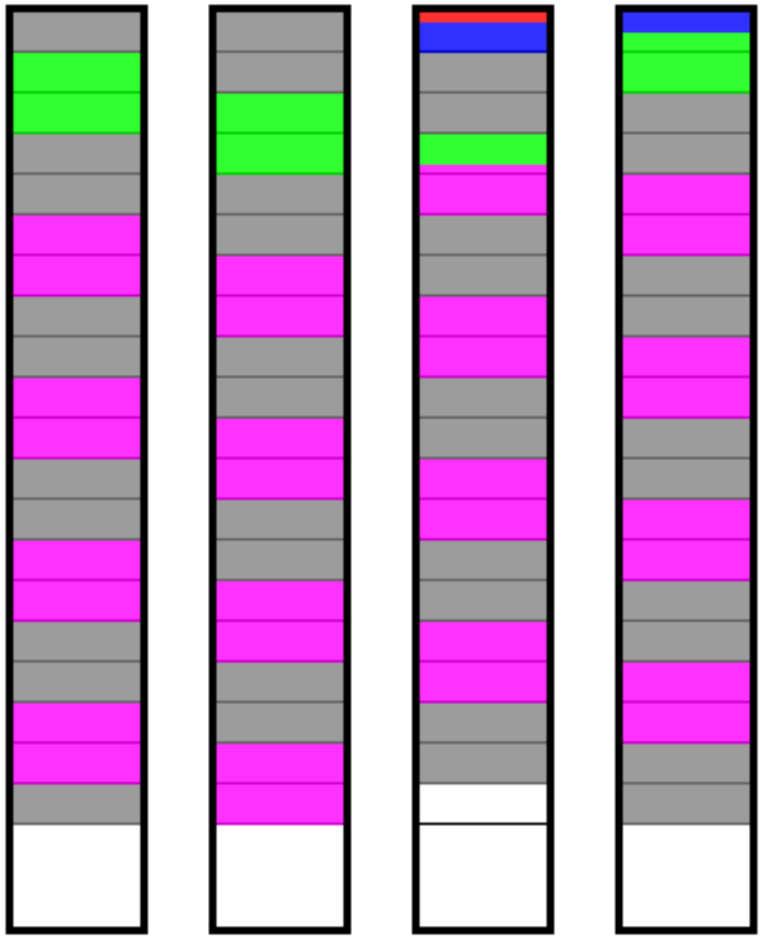
Block-interleaved
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Level 5



Block-interleaved
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Level 6



Block-interleaved
The red file: 4KB
The blue file: 20KB
The green file: 100KB
The magenta file: 500 KB

RAID Recap

Category	Level	Description	I/O Request Rate (Read/Write)	Data Transfer Rate (Read/Write)	Typical Application
Striping	0	Nonredundant	Large strips: Excellent	Small strips: Excellent	Applications requiring high performance for noncritical data
Mirroring	1	Mirrored	Good/Fair	Fair/Fair	System drives; critical files
Parallel access	2	Redundant via Hamming code	Poor	Excellent	
	3	Bit-interleaved parity	Poor	Excellent	Large I/O request size applications, such as imaging, CAD
Independent access	4	Block-interleaved parity	Excellent/Fair	Fair/Poor	
	5	Block-interleaved distributed parity	Excellent/Fair	Fair/Poor	High request rate, read-intensive, data lookup
	6	Block-interleaved dual distributed parity	Excellent/Poor	Fair/Poor	Applications requiring extremely high availablity

Highlights

- The six RAID organizations
- RAID-1, RAID-3 and RAID-5 are the most interesting