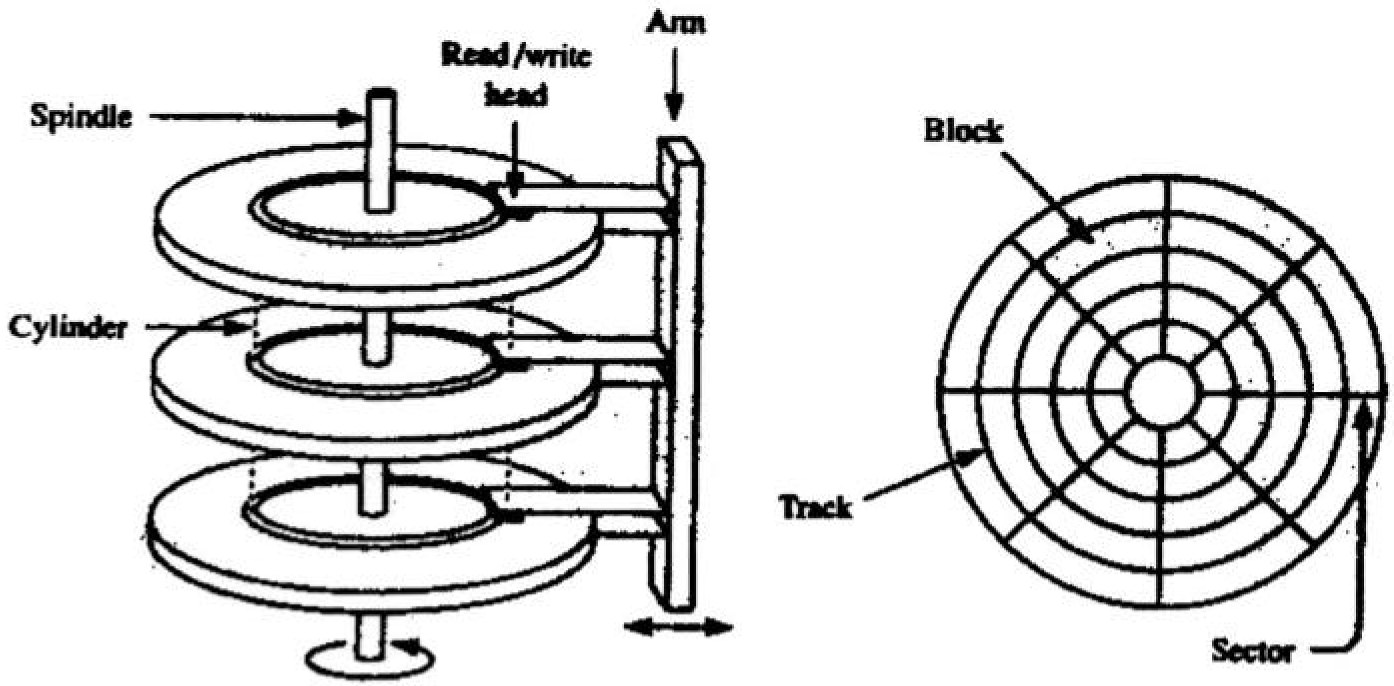
Part IV I/O System

Chapter 12: Mass Storage Structure

Disk Structure

- Three elements: cylinder, track and sector/block.
- Three types of latency (*i.e.*, delay)
  - Positional or seek delay – mechanical and slowest
  - Rotational delay
  - Transfer Delay



Computing Disk Latency

- **Track size: 32K = 32,768 bytes**
- **Rotation Time: 16.67 msec (millisecond)**
- **Average seek time: 30 msec**
- **What is the average time to transfer *k* bytes?**

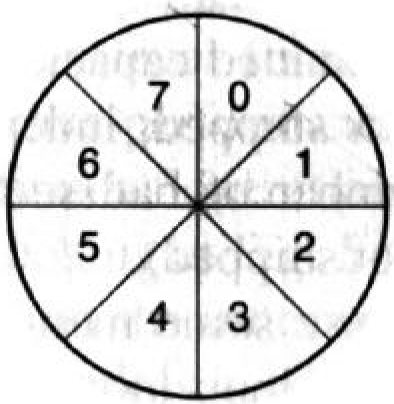
Average read time = 30 + 16.67/2 + (*k*/32K)´ 16.67

average time to move from track to track (→ 30)

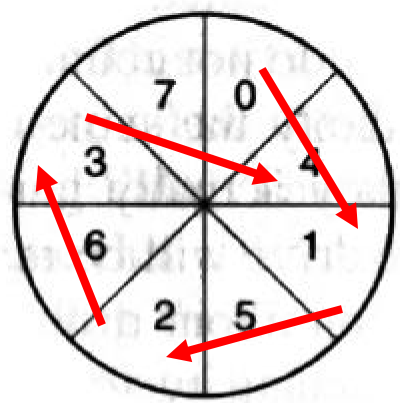
on average, wait a half turn (→ 16.67/2)

this is the "length" the disk head must pass to complete a transfer (→ (k/32K))

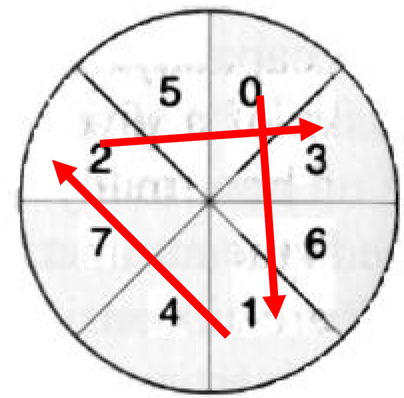
Disk Block Interleaving



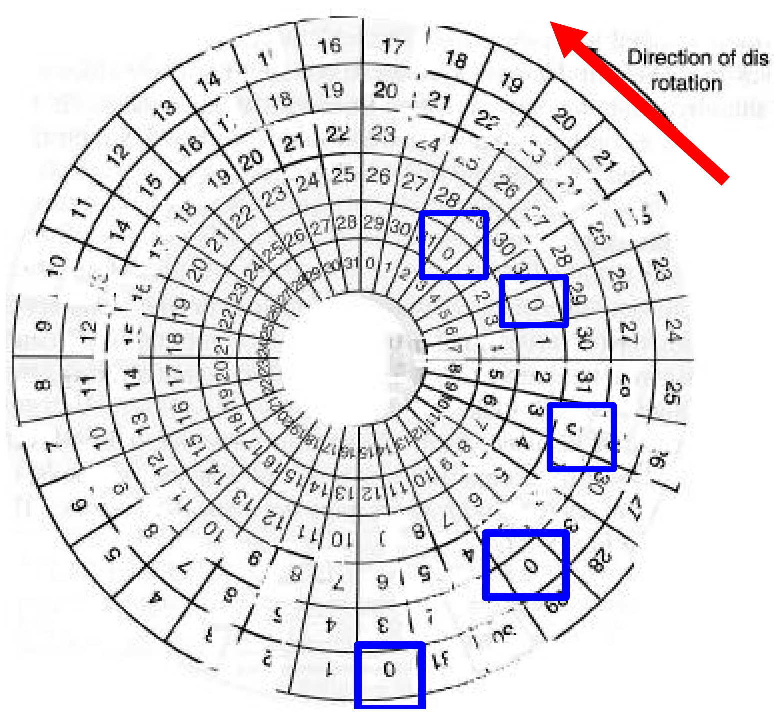
no interleaving



single interleaving



double interleaving



Cylinder Skew

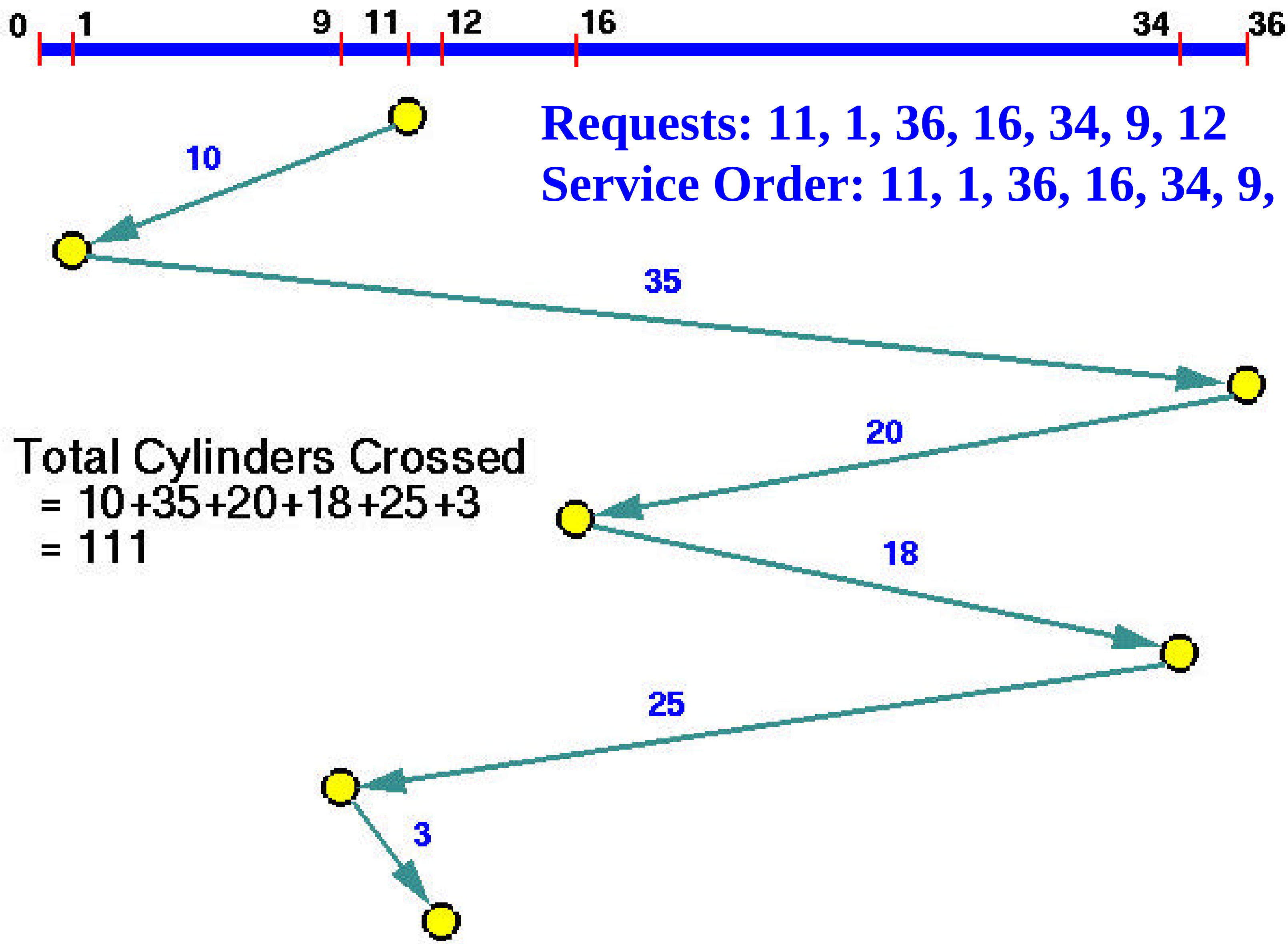
The position of sector/block 0 on each track is offset from the previous one, providing sufficient time for moving the disk head from track to track.

Disk Scheduling

- Since seeking is time consuming, disk scheduling algorithms try to minimize this latency.
- We shall discuss the following algorithms:
  - First-come, first served (FCFS)
  - Shortest-seek-time-first (SSTF)
  - SCAN and C-SCAN
  - LOOK and C-LOOK
- Since seeking only involves cylinders, the input to these algorithms are cylinder numbers.

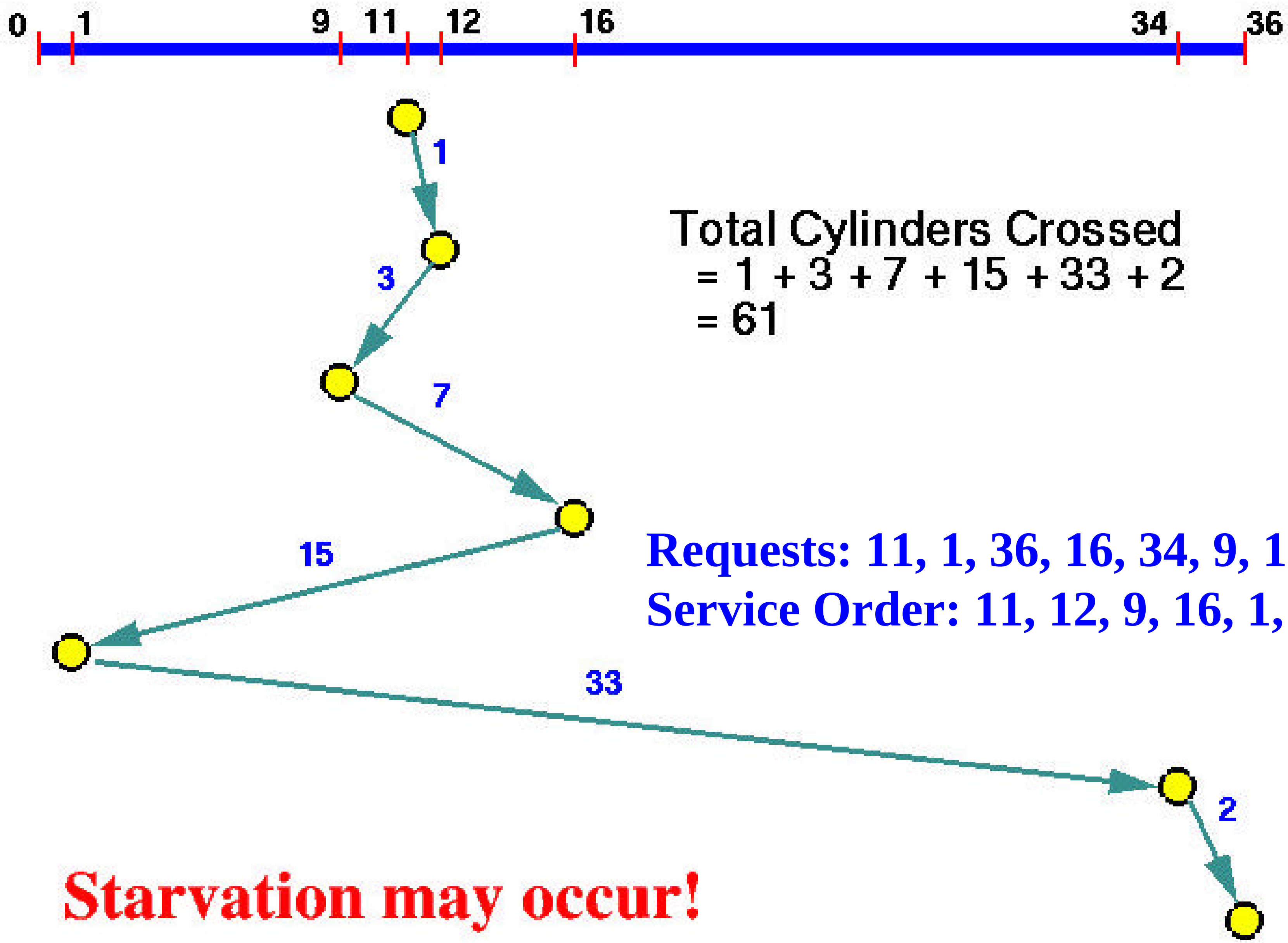
First-Come, First-Served

Requests: 11, 1, 36, 16, 34, 9, 12
Service Order: 11, 1, 36, 16, 34, 9, 12



Total Cylinders Crossed
= 10+35+20+18+25+3
= 111

Shortest-Seek-Time-First



Total Cylinders Crossed
= 1 + 3 + 7 + 15 + 33 + 2
= 61

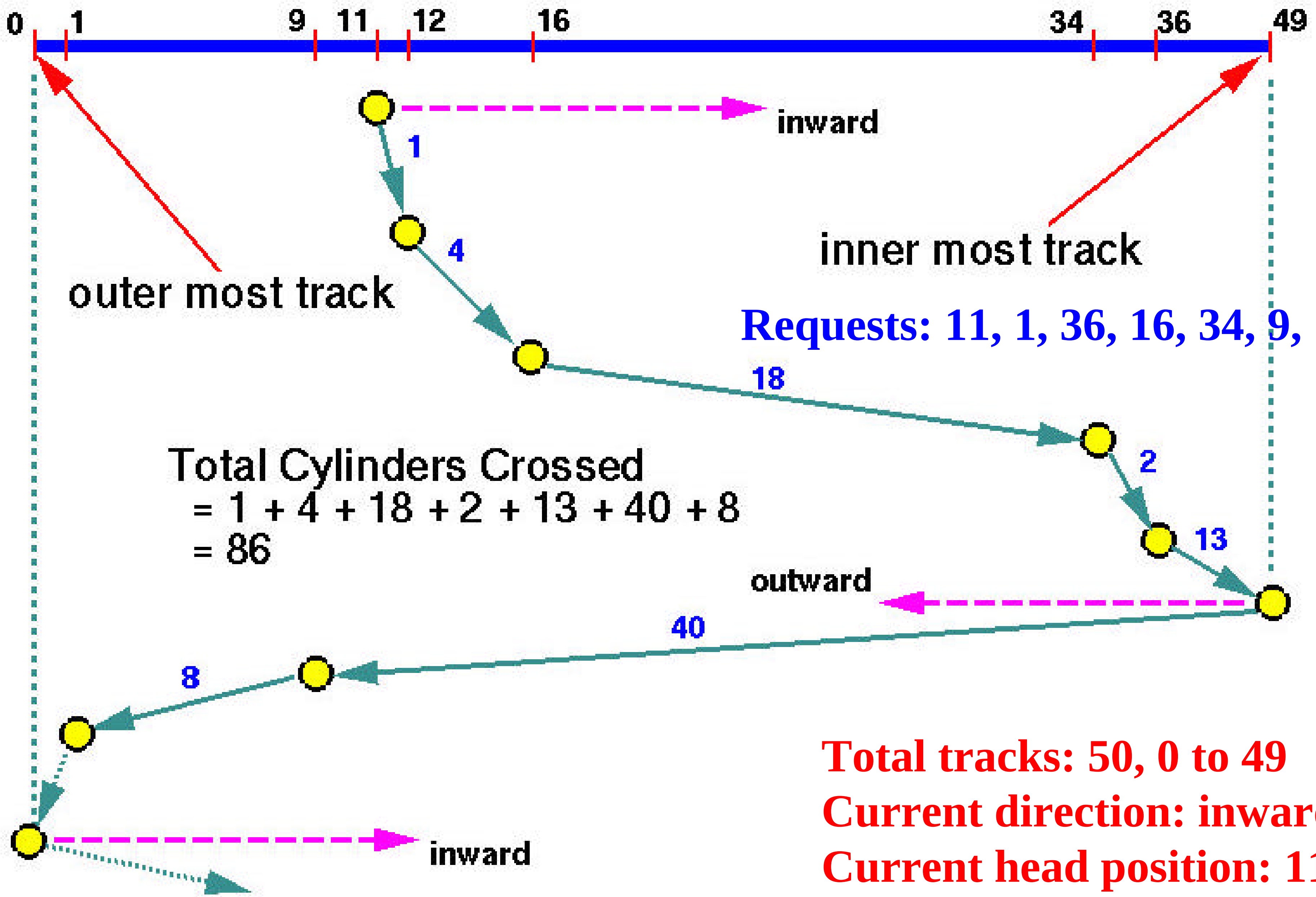
Requests: 11, 1, 36, 16, 34, 9, 12
Service Order: 11, 12, 9, 16, 1, 34, 36

Starvation may occur!

SCAN Scheduling: 1/2

- **This algorithm requires one more piece of information: the disk head movement direction, *inward* or *outward*.**
- **The disk head starts at one end, and move toward the other in the *current* direction.**
- **At the other end, the direction is *reversed* and service continues.**
- **Some authors refer the SCAN algorithm as the elevator algorithm. However, to some others the elevator algorithm means the LOOK algorithm.**

SCAN Scheduling: 2/2



Requests: 11, 1, 36, 16, 34, 9, 12

Total tracks: 50, 0 to 49

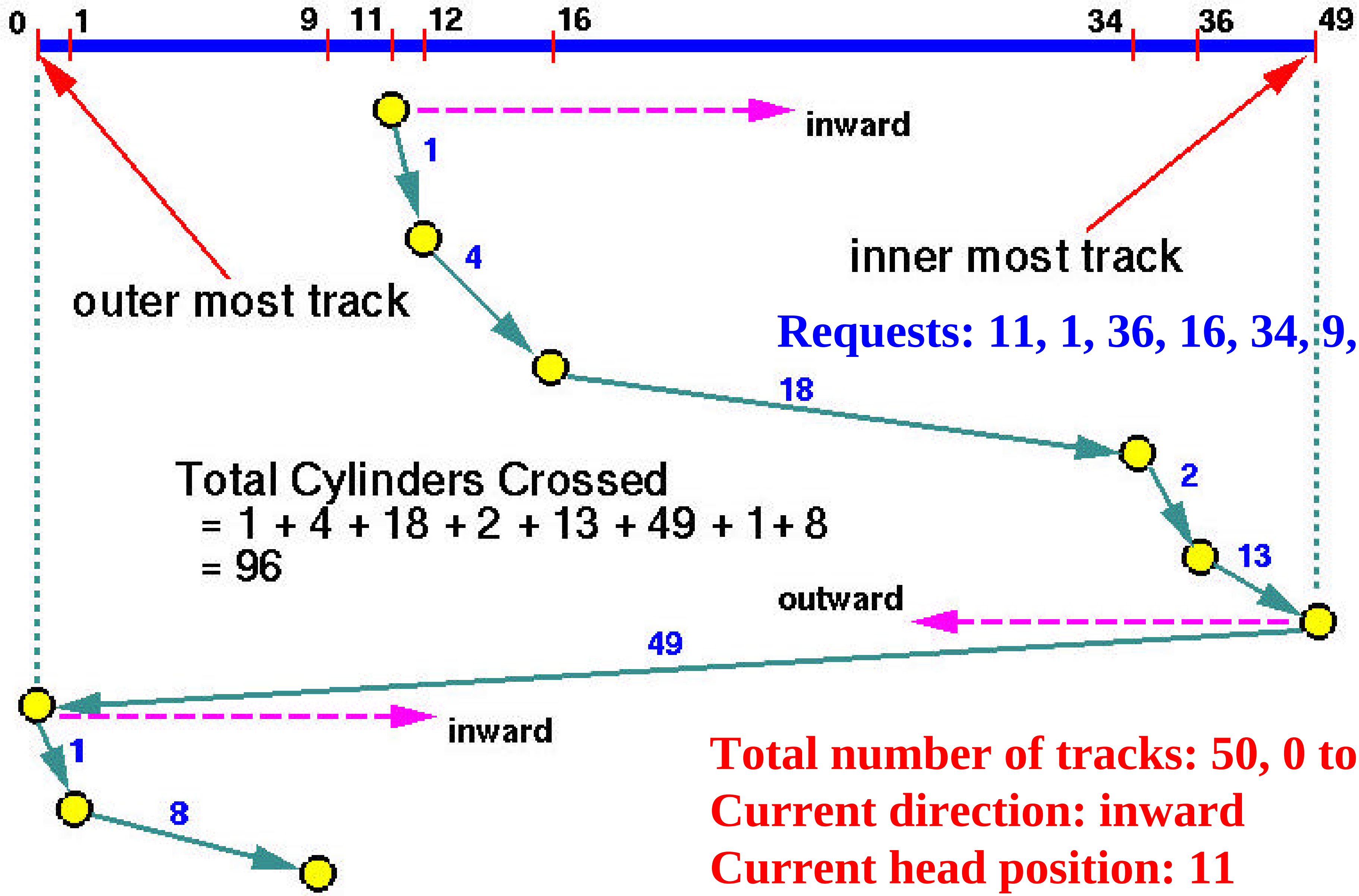
Current direction: inward

Current head position: 11

C-SCAN Scheduling: 1/2

- C-SCAN is a variation of SCAN.
- When the disk head reaches one end, move it back to the other end. Thus, this is simply a wrap-around.
- The C in C-SCAN means circular.

C-SCAN Scheduling: 2/2



Requests: 11, 1, 36, 16, 34, 9, 12

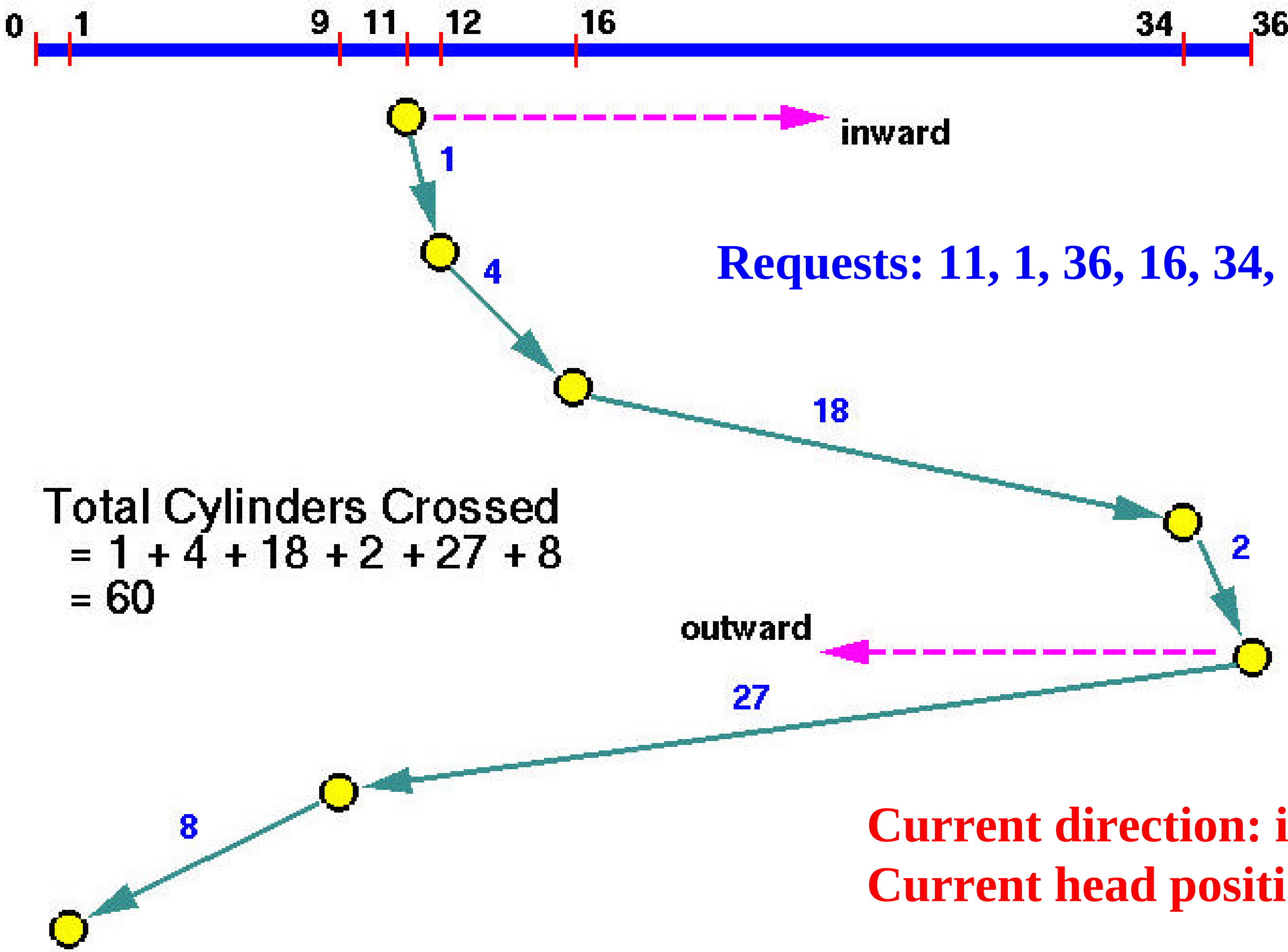
Total Cylinders Crossed
= 1 + 4 + 18 + 2 + 13 + 49 + 1 + 8
= 96

Total number of tracks: 50, 0 to 49
Current direction: inward
Current head position: 11

LOOK Scheduling: 1/2

- **With SCAN and C-SCAN, the disk head moves across the full width of the disk.**
- **This is very time consuming. In practice, SCAN and C-SCAN are not implemented this way.**
- **LOOK: It is a variation of SCAN. The disk head goes as far as the last request and reverses its direction.**
- **C-LOOK: It is similar to C-SCAN. The disk head also goes as far as the last request and reverses its direction.**

LOOK Scheduling: 2/2

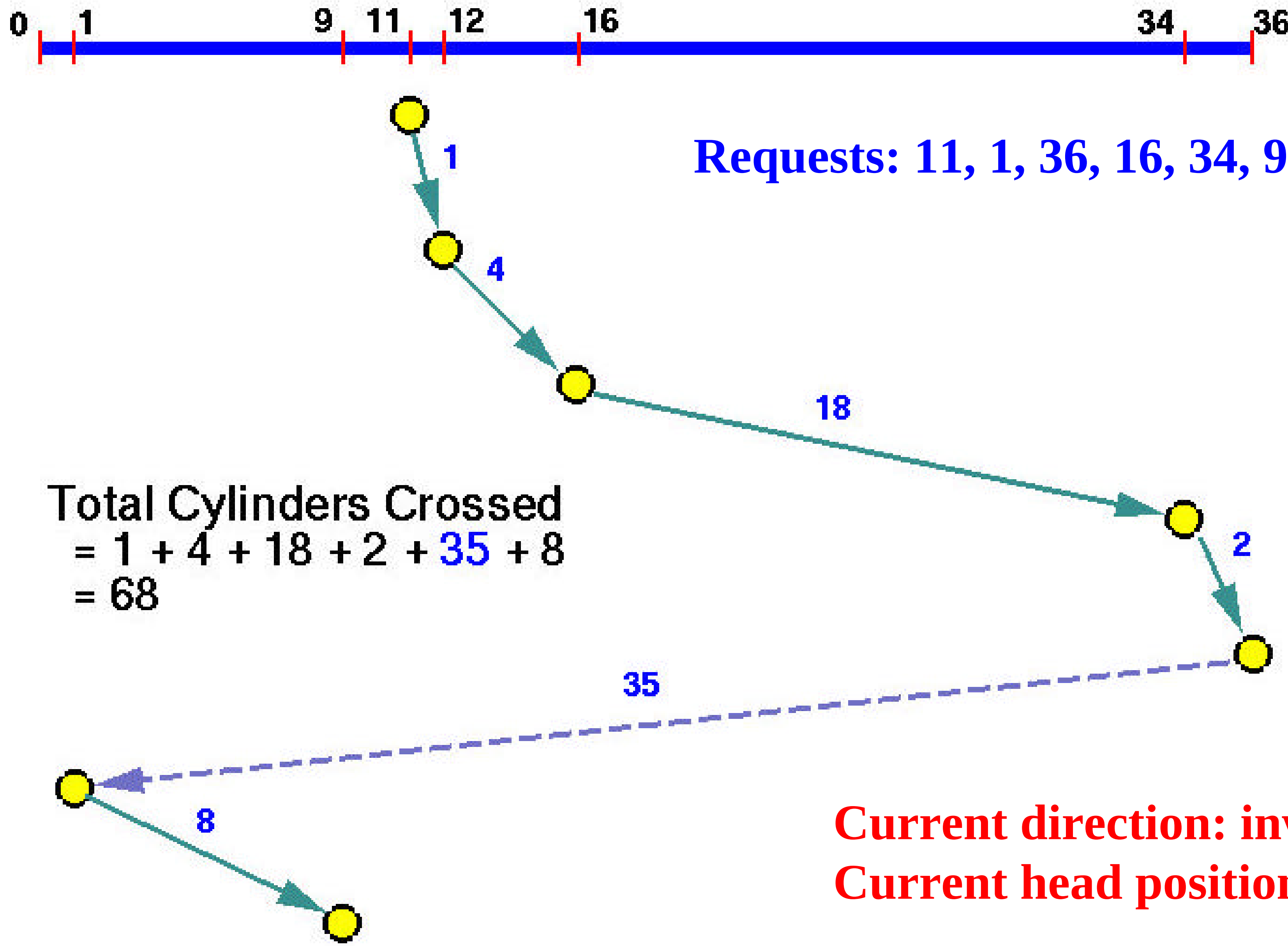


Requests: 11, 1, 36, 16, 34, 9, 12

Total Cylinders Crossed
= 1 + 4 + 18 + 2 + 27 + 8
= 60

Current direction: inward
Current head position: 11

C-LOOK Scheduling



Requests: 11, 1, 36, 16, 34, 9, 12

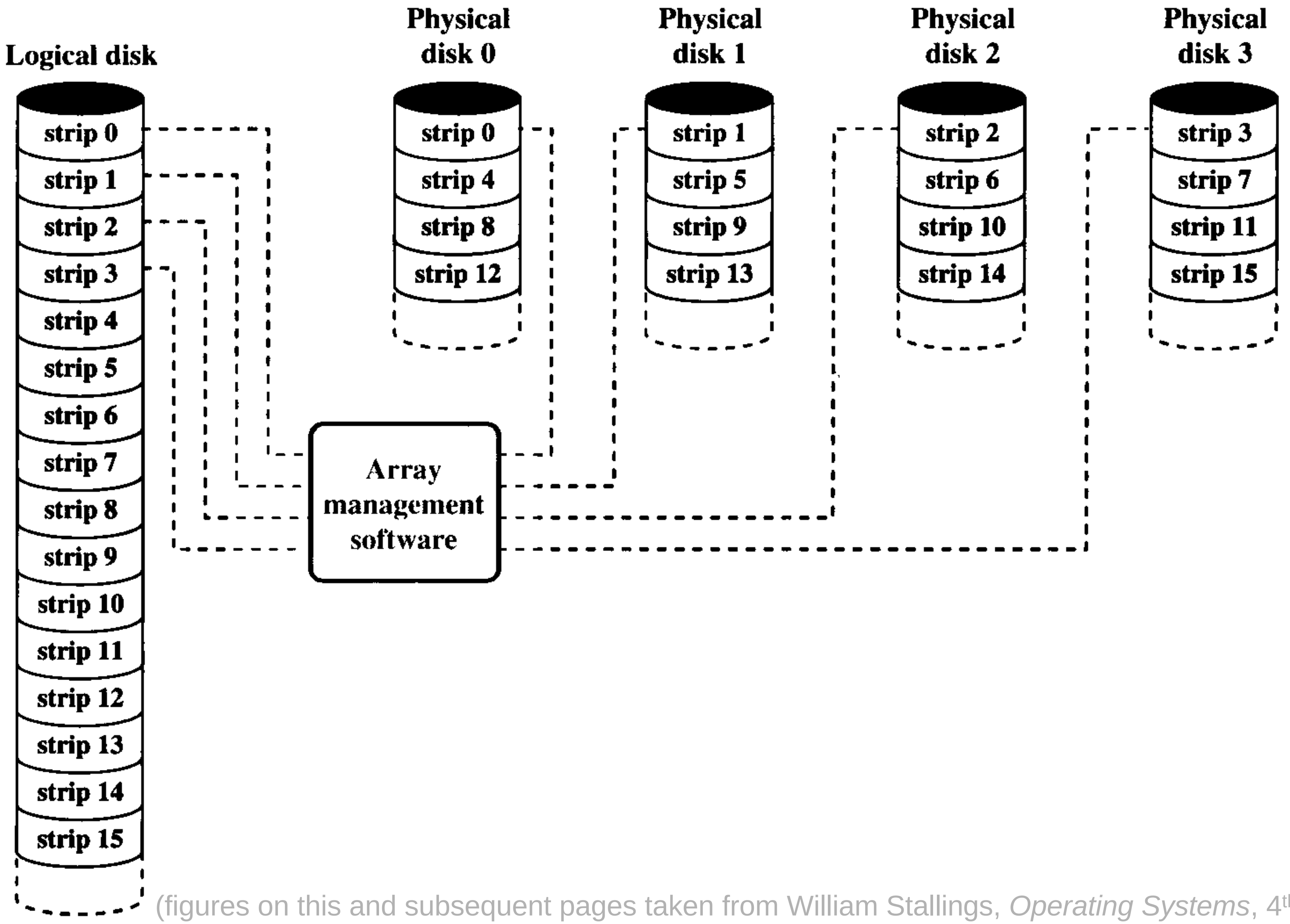
Total Cylinders Crossed
= 1 + 4 + 18 + 2 + 35 + 8
= 68

Current direction: inward
Current head position: 11

RAID Structure: 1/2

- **RAID: Redundant Arrays of Inexpensive Disks.**
- **RAID is a set of physical drives viewed by the operating system as a single logical drive.**
- **Data are distributed across the physical drivers of an array.**
- **Redundant disk capacity is used to store parity information, which guarantees data recoverability is case of disk failure.**
- **RAID has 6 levels, each of which is not necessary an extension of the other.**

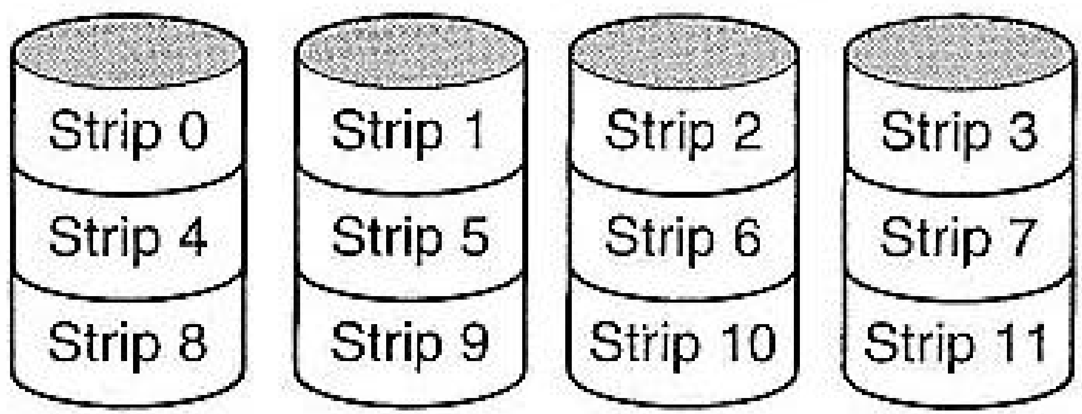
RAID Structure: 2/2



(figures on this and subsequent pages taken from William Stallings, *Operating Systems*, 4th ed)

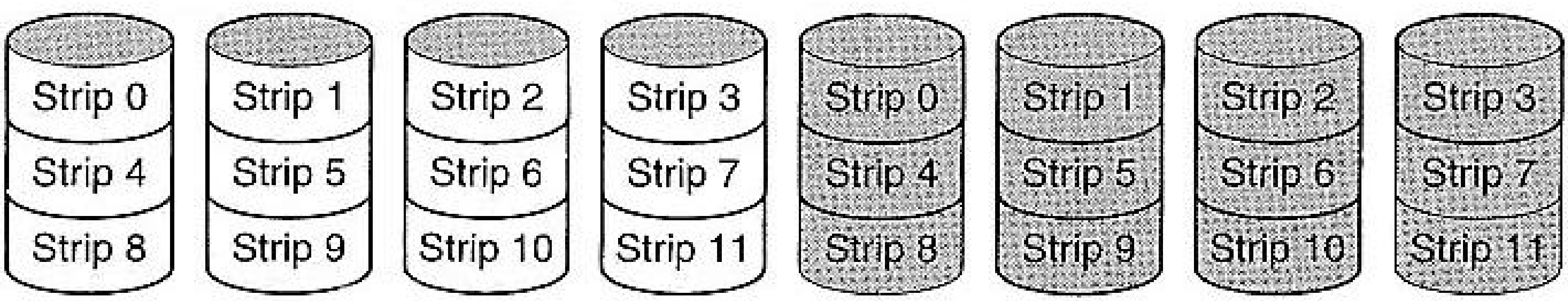
RAID Level 0

- The virtual single disk simulated by RAID is divided up into strips of ***k*** sectors each.
- Consecutive strips are written over the drivers in a round-robin fashion. There is no redundancy.
- If a single I/O request consists of multiple contiguous strips, then multiple strips can be handled in parallel.



RAID Level 1: Mirror

- Each logical strip is mapped to two separate physical drives so that every drive in the array has a mirror drive that contains the same data.
- Recovery from a disk failure is simple due to redundancy.
- A write request involves two parallel disk writes.
- Problem: Cost is high (doubled)!

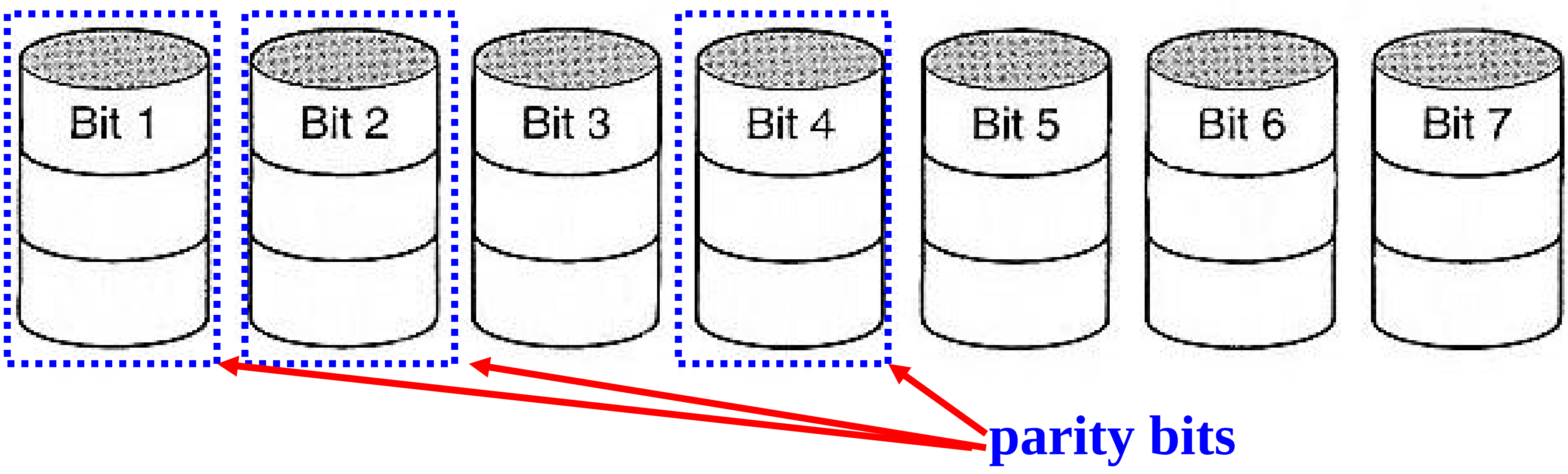


Parallel Access

- **RAID Levels 2 and 3 require the use of *parallel access technique*. In a parallel access array, all member disks participate in the execution of every I/O and the spindles of the individual drives are synchronized so that each disk head is in the same position on each disk at any given time.**
- **Data strips are very small, usually a single byte or word.**

RAID Level 2: 1/2

- An error-correcting code is calculated across corresponding bits on each data and the bits of code are stored in the corresponding bit positions on disks.
- Example: A 8-bit byte is divided into two 4-bit nibbles. A 3-bit Hamming code is added to form a 7-bit word, of which bits 1, 2 and 4 are parity bits. This 7-bit Hamming coded word is written to the seven disks, one bit per disk.

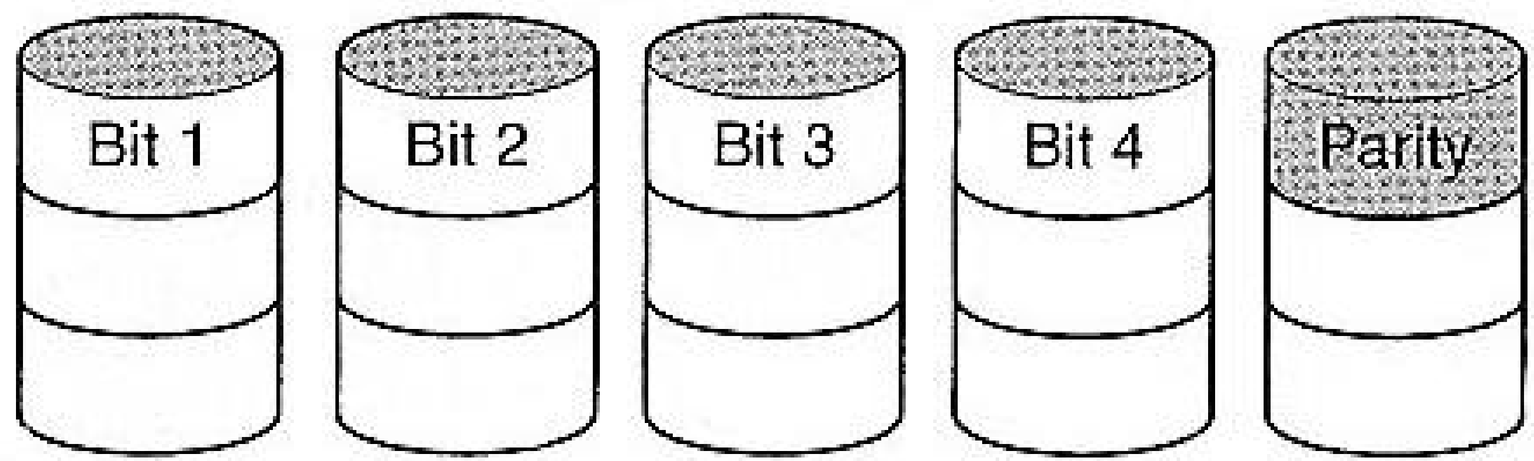


RAID Level 2: 2/2

- Cost is high, although the number of bits needed is less than that of RAID 1 (mirror).
- The number of redundant disks is $O(\log_2 n)$, where n is the number of data disks.
- On a single read, all disks are accessed at the same time. The requested data and the associated error-correcting code are delivered to the controller. If there is error, the controller reconstructs the data bytes using the error-correcting code. Thus, read access is not slowed.
- RAID 2 would only be an effective choice in the environment in which many disk errors occur.

RAID Level 3: 1/2

- RAID 3 is a simplified version of to RAID 2. It only needs one redundant drive.
- A single parity bit is computed for each data word and written to a **parity drive**.
- **Example**: The parity bit of bits 1-4 is written to the same position on the parity drive.

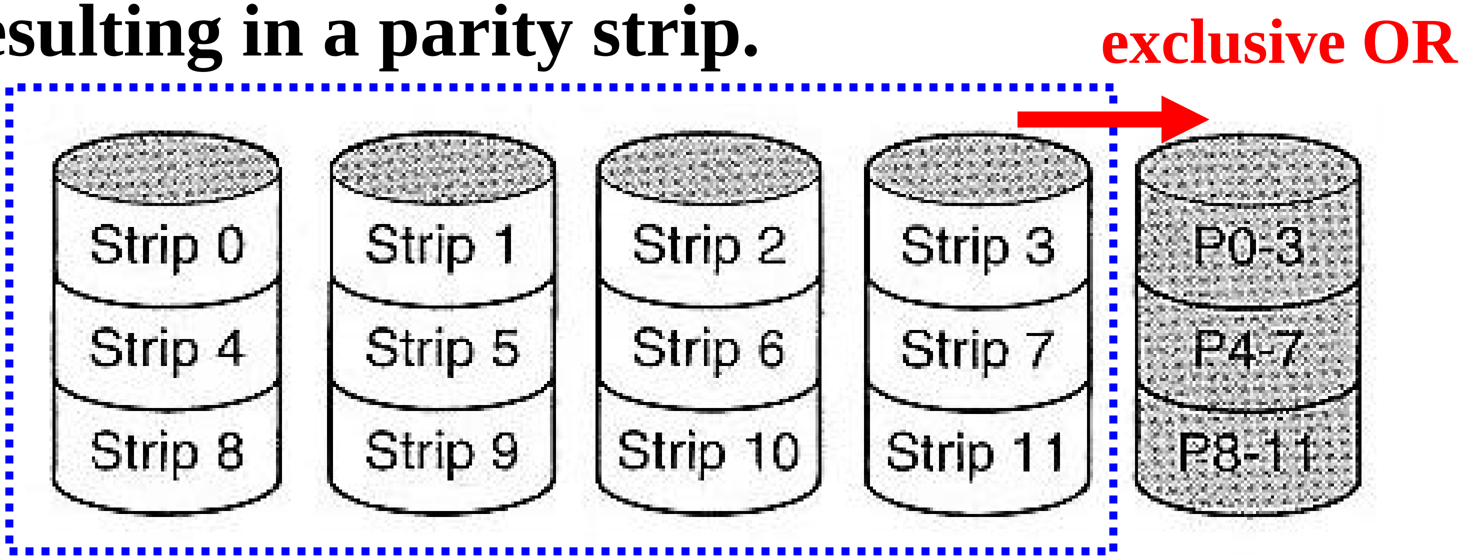


RAID Level 3: 2/2

- If one failing drive is known, the parity bit can be used to reconstruct the data word.
- Because data are striped in very small strips, RAID 3 can achieve very high data transfer rates.
- Any I/O request will involve the parallel transfer of data from all of the data disks. However, only one I/O request can be executed at a time.

RAID Level 4: 1/2

- RAID 4 and RAID 5 work with strips rather than individual data words, and do not require synchronized drives.
- The parity of all strips on the same “row” is written on an parity drive.
- Example: strips 0, 1, 2 and 3 are exclusive-Ored, resulting in a parity strip.



RAID Level 4: 2/2

- If a drive fails, the lost bytes can be reconstructed from the parity drive.
- If a sector fails, it is necessary to read *all* drives, including the parity drive, to recover.
- The load of the parity drive is very heavy.

RAID Level 5

- To avoid the bottleneck of the parity drive of RAID 4, the parity strips can be distributed uniformly over all drives in a round-robin fashion.
- However, data recovery from a disk crash is more complex.