COMPS267F Chapter 10

IO Structure and Disk Management

Dr. Andrew Kwok-Fai LUI

Learning Objective @ Chapter 10

- Explain the issue of diversity of IO devices
- Explain how a layered IO structure resolves the problem of IO device diversity

io management

IO Devices

- Many IO devices can be connected to computer
- The devices are differentiated in these aspects
  - Unit of data transfer
    - Character stream or block transfer
  - Interaction of programs and IO operations
    - Synchronous or Asynchronous
  - Nature of data access
    - Sequential or random
  - IO device mutual exclusion
    - Sharable or dedicated
  - Speed of IO device
  - Read and write

Aim of IO Management

- Hiding the great disparity of hardware configuration from the user
  - The approach is to present a uniform interface to the user,
  - Carrying out the necessary mapping of instructions and data structure to the underlying hardware

Concept of Device Independence

- A uniform method or interface exists for the manipulation of very different underlying hardware
  - Not really a new concept
  - Same write operation to write a file to CDROM, hard disk, and floppy disk
  - Plug-and-play is an up-to-date example of this concept
    - A new hardware should be simply added to a computer without the need to re-configure
    - The OS should be designed to carry out the necessary management and mapping
  - Uniform naming
    - In UNIX, a particular pathname may refer to a file or a device

Plug and Play

- The concept of plug-and-play is often associated with Microsoft range of operating systems (from Windows 95 onwards)
  - Actually implemented by a number of other earlier operating systems

Plug and Play

- Connecting a piece of hardware to a computing can be hazardous
  - Two electrical circuitries are connected
  - Physically and electrically safe is the first condition of a plug-and-play system.
- <u>Hot-swapping</u> plug-and-play allows plugging and unplugging of new hardware without risk of damage
- Self-configuration is the ability to configure a piece of hardware without user intervention or any need for software configuration programs.

Plug and Play

- Each piece of hardware must provide an ID so that it can be recognized
- The OS must be designed to handle the detection and handling of new hardware
  - Plugging in causes an interrupt
  - OS reads from the bus information about the new hardware
  - Determine and load in the appropriate device driver
  - Network connectivity allows the download of device driver
  - Older OS would need to spend longer time to scan the whole system for changes in hardware

Example: Universal Serial Bus (USB)

- A USB source can be connected to a multitude of USB devices (up to 127 devices) through branching.
  - The branching of a USB connection happens in a USB hub
- Hot-swappable
  - New device can be plug-in without the need to restart the computer
- Multiple transfer speeds
  - 1.5Mbs for human interface devices
  - 12Mbs for full speed transfer
  - 480Mbs for USB 2.0
  - 20Gbs for USB 3.2

Example: Universal Serial Bus (USB)

- A number of device classes are defined for USB devices
  - An OS should implement all the device classes

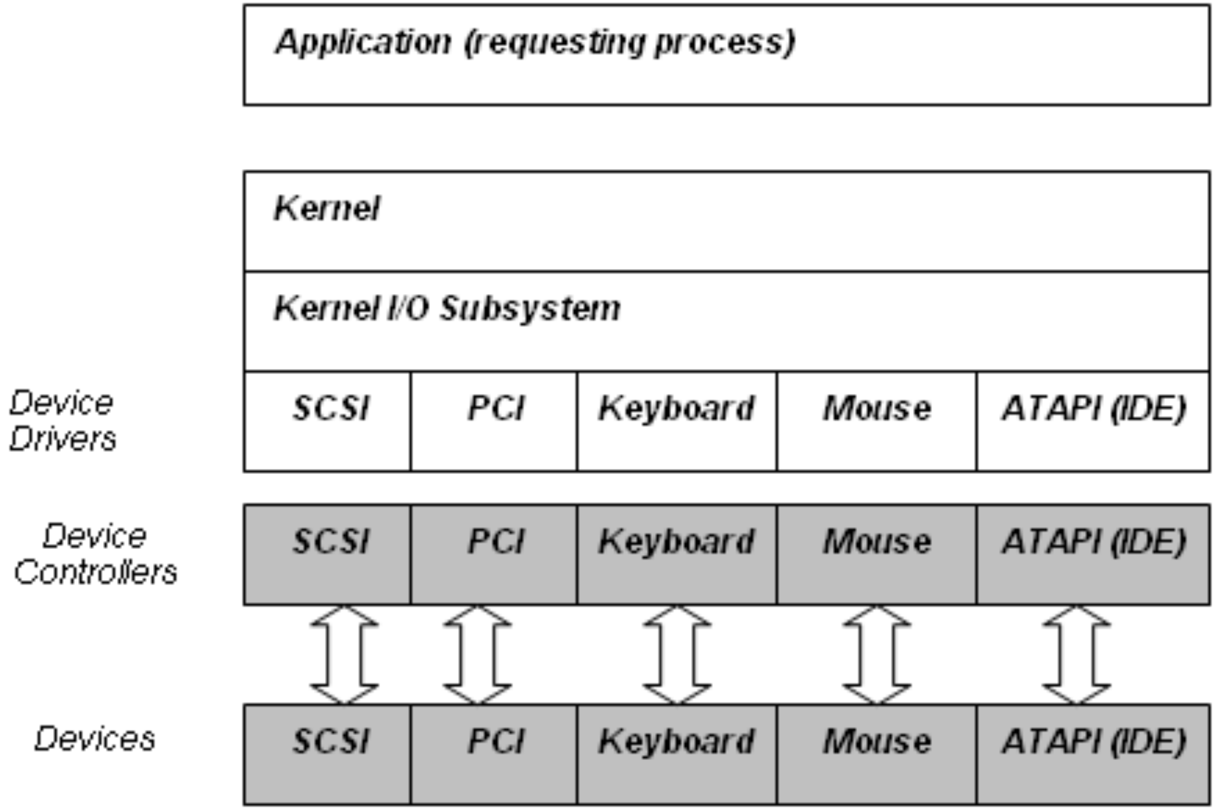
Device ID	*Device Class*
0	Reserved
1	Audio devices such as sound card
3	Human interface such as keyboard, mouse, etc
6	Camera uses Picture Transfer Protocol
7	Printer devices
8	Mass storage device. Device is presented as a file system
9	USB Hubs
10	Communications devices such as wireless network interface
14	Video devices, webcam
224	Wireless controllers such as Bluetooth
255	Custom

io system structure

IO System Structure

- The objective is to offer the application (user) a uniform I/O interface to different devices
  - Layered structure

IO System Structure



IO System Structure

- The layered structure aims to tackle the diversity of device characteristics
  - The lower layers deal with all the complexities of real devices
  - The upper layer provides an uniform interface
  - A device driver is a software module that provides a device-independent interface for the kernel I/O subsystem
    - A device driver for every type of device
  - The devices are characterized using some standard interface
    - Leave the complexity out of the kernel

Kernel IO System Sub-System

- A system manager translating user requests into low level I/O requests
  - Allocating system resources (like buffers) for these requests
  - Sending the requests to the appropriate device driver
    - Device driver maps these generic I/O requests into device-specific instructions for the device controllers
  - The hardware executes the commands
  - The interrupt handler handles the returned status

Kernel IO System Sub-System

- Providing four IO related services
  - I/O Scheduling
    - Order and re-order IO request for maximum performance
  - Buffering
    - Rectifying difference of speed between computer and IO device
  - Caching
    - Keeping essential information such as metadata
  - Error Handling

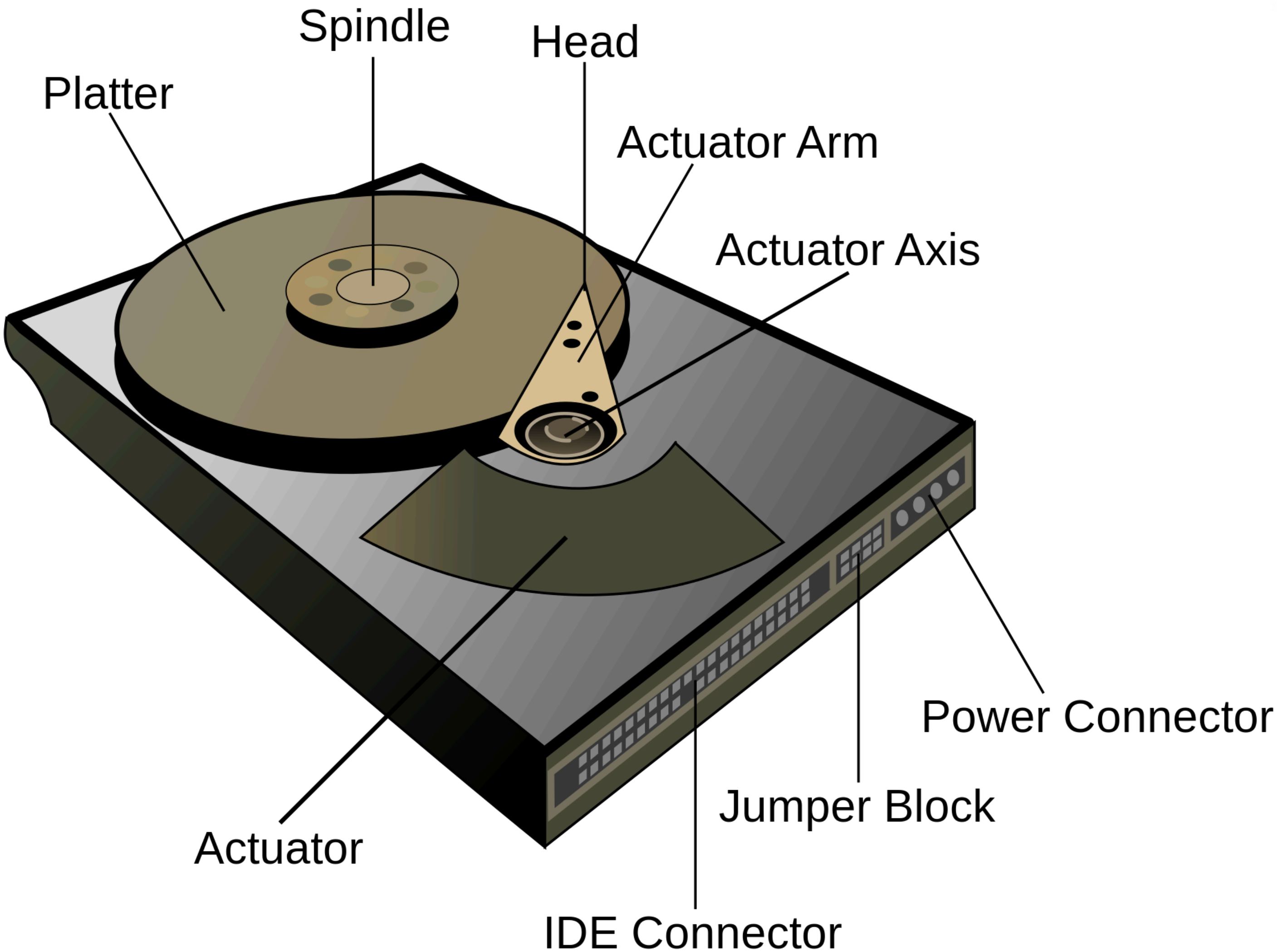
Learning Objective @ Chapter 10

- Explain the operation of common disk scheduling algorithms
- Analyze the performance of disk scheduling algorithms

case study: disk management

Hard Disks

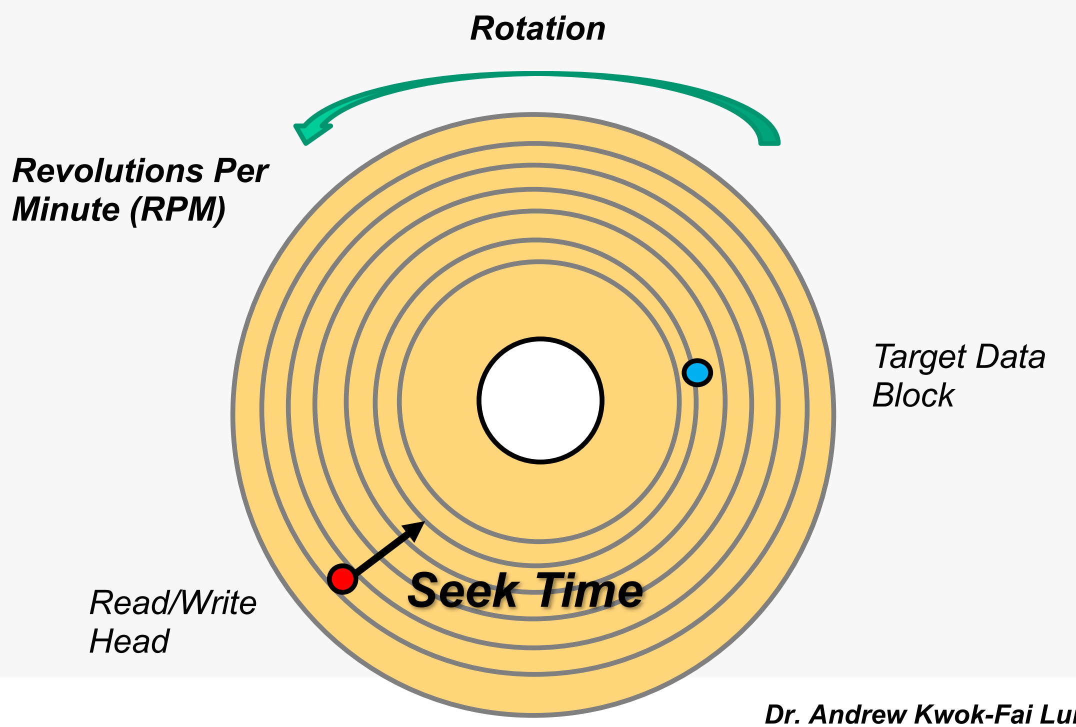
- Hard disks provide the secondary storage for modern computers
  - Mechanical hard disks are still common because of their sizes
  - Gives good file system performance
  - The disk space is addressed as 1D array of blocks
    - Logical block addressing (LBA)
    - A logical block is the smallest unit of transfer
  - Previously, disks were addressed as a set of cylinder, track, and section number



Source: Wikipedia

Disk Scheduling

- Selection of a request from the pending disk I/O requests so that the overall seek time is minimized
- The time taken in mechanical disk operations is contributed by the following components
  - Seek time
  - Rotational latency
  - Data transfer time



Dr. Andrew Kwok-Fai Lui

Disk Scheduling Algorithms

- Disk scheduling algorithms are different ways to schedule the handling of disk requests

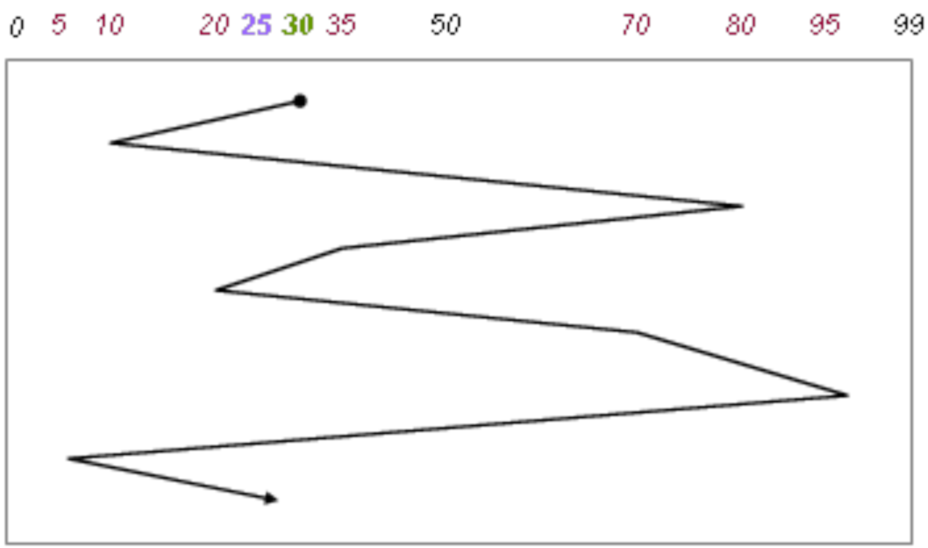
Disk Scheduling Algorithms: FCFS

- First-Come First-Served Scheduling (FCFS)
  - Scheduling the disk requests based on their order of arrival
  - May result in lots of disk head movement
  - For example, if one request is for a track near the centre and the next request is for a track near the edge of the disk, then seek time will be very high

Disk Scheduling Algorithms: FCFS

First-Come-First-Served (FCFS)

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30



Tracks Travelled = (30 – 10) + (80 – 10) + (80 – 35) + (35 – 20) + (70 – 20) + (95 – 70) + (95 – 5) + (25 – 5)

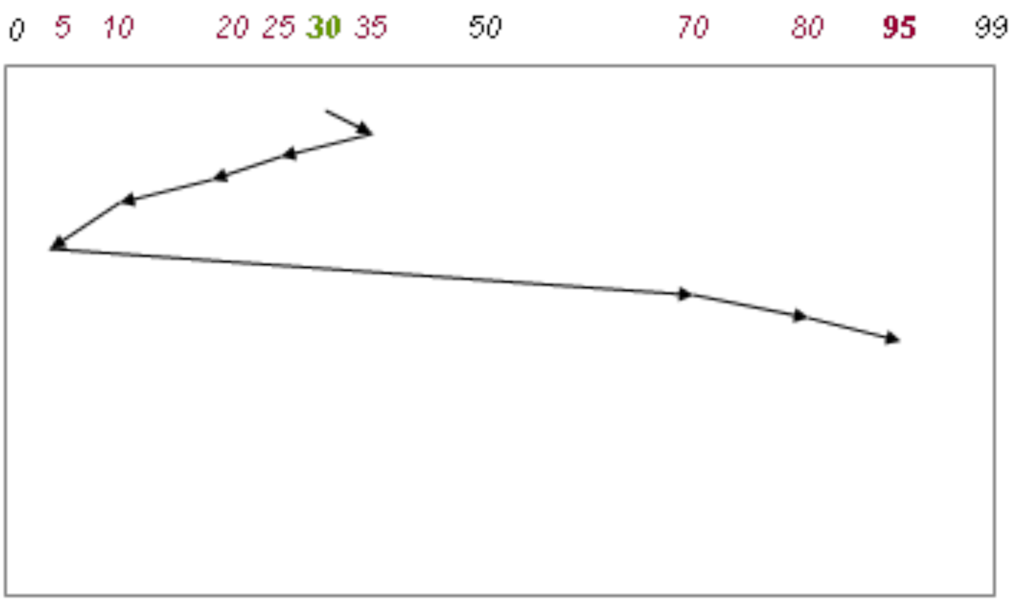
Disk Scheduling Algorithms: SSTF

- Shortest-Seek-Time-First (SSTF) scheduling
  - Servicing all requests close to the current head position first
  - More requests can be satisfied before moving the disk head far away to service other requests.
  - May result in starvation for some requests that are furthest away from the current head position

Disk Scheduling Algorithms: SSTF

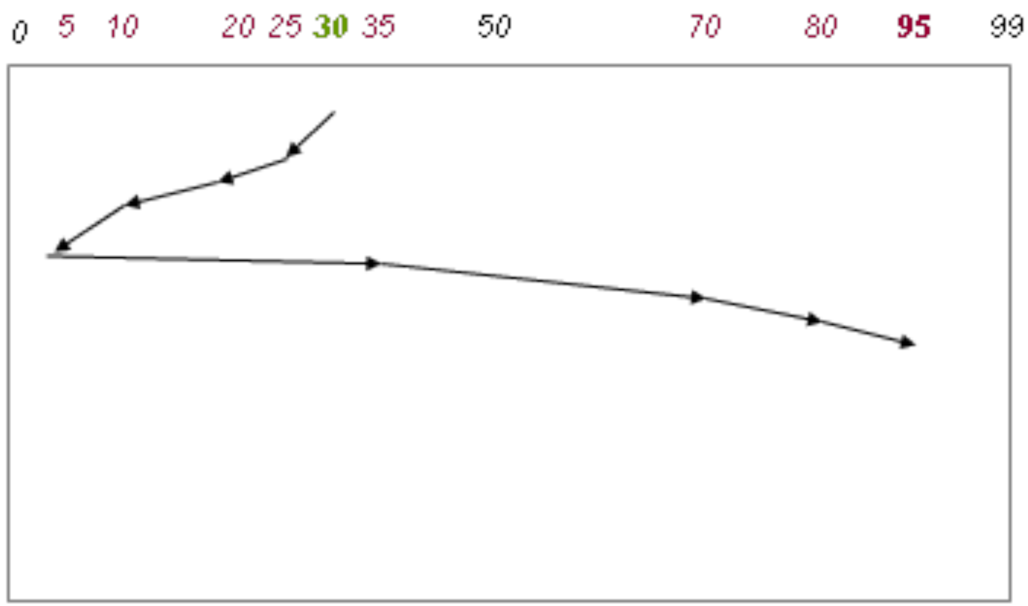
Shortest-Seek-Time-First (SSTF)

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30



Tracks Travelled = (35 – 30) + (35 – 5) + (95 – 5)

Disk Scheduling Algorithms: SSTF



Tracks Travelled = (30 - 5) + (95 – 5)

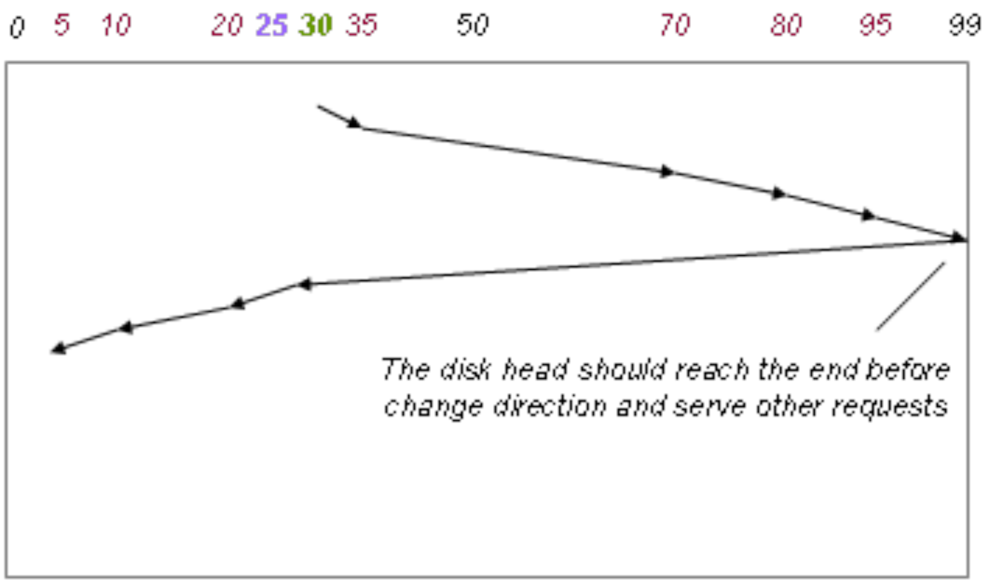
Disk Scheduling Algorithms: SCAN

- SCAN scheduling works like an elevator
  - Organizing the requests that the disk head sweeps from one end of the disk to the other it services all the requests on the way.
  - Handles any requests on its way back as well
  - Requests that have just missed the disk head will have to wait a long time (but bounded).
    - Particularly for those requests near one side of the disk.

Disk Scheduling Algorithms: SCAN

SCAN

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30 (Travelling from Low to High)



Tracks Travelled = (99 – 30) + (99 – 5)

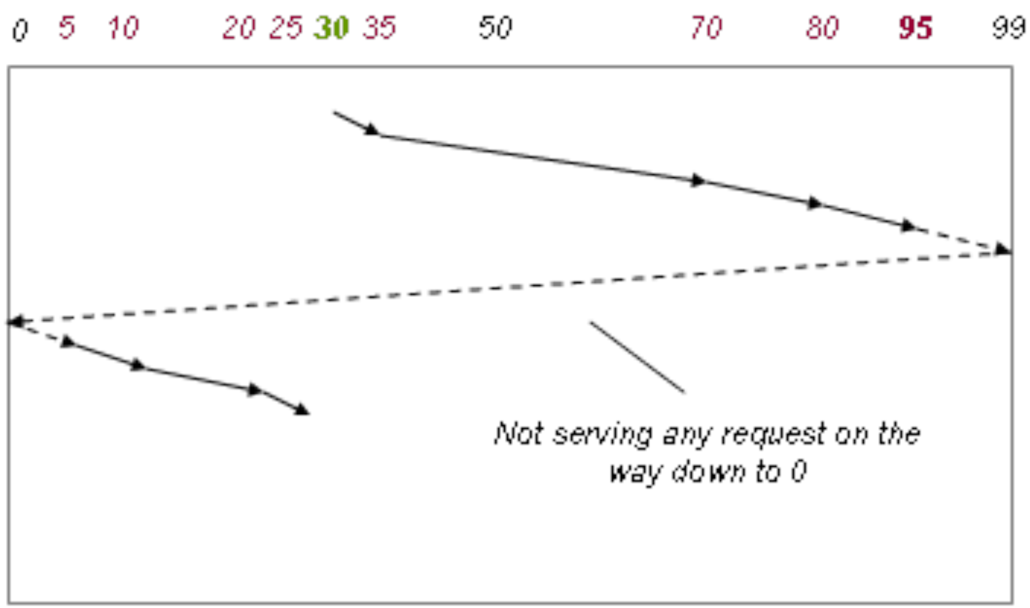
Disk Scheduling Algorithms: C-SCAN

- C-SCAN is SCAN scheduling with a minor variation.
  - Instead of servicing requests on the way back the disk head goes back immediately to its starting point.
  - An elevator going from ground floor to the top floor, and then heads directly down to ground floor without taking any more passengers on its way down
  - More uniform waiting time

Disk Scheduling Algorithms: C-SCAN

C-SCAN (Service from Low to High)

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30 (Travelling from Low to High)



Tracks Travelled = (99 – 30) + (99 – 0) + (25 – 0)

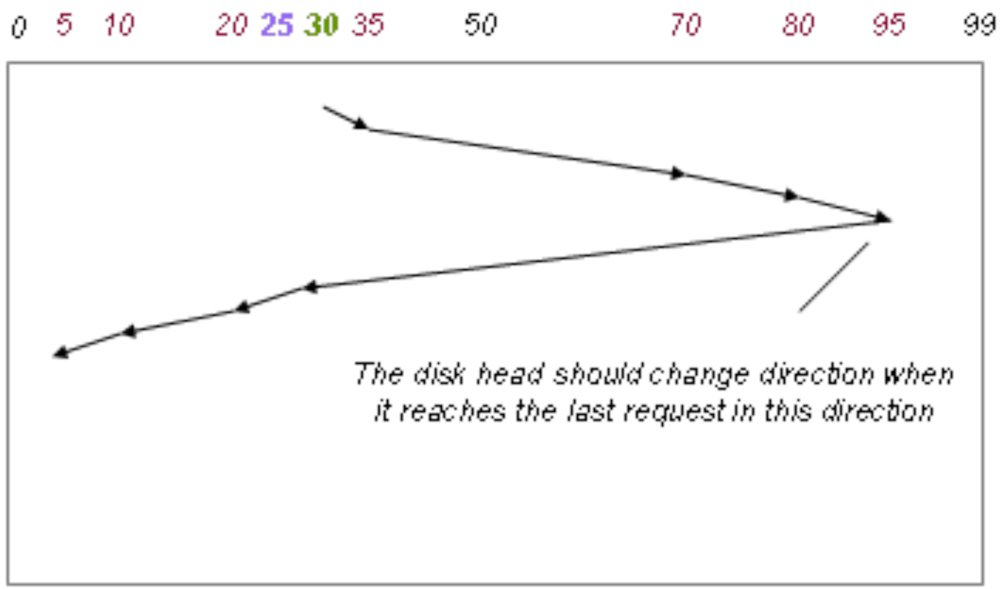
Disk Scheduling Algorithms: LOOK/C-LOOK

- A minor variation of SCAN/C-SCAN scheduling
  - Disk head will stop at the track of the last request in each direction and then change direction
  - C-LOOK is LOOK scheduling that serve only in one direction

Disk Scheduling Algorithms: LOOK/C-LOOK

LOOK

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30 (Travelling from Low to High)

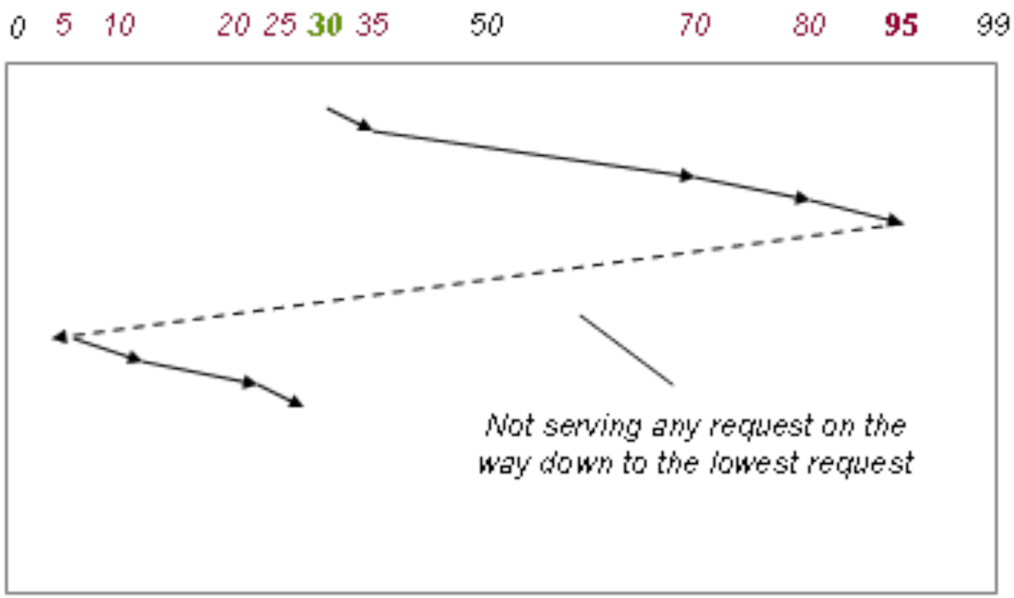


Tracks Travelled = (95 – 30) + (95 – 5)

Disk Scheduling Algorithms: LOOK/C-LOOK

C-LOOK (Service from Low to High)

Pending Requests:	10 80 35 20 70 95 5 25
Disk Head Position:	30 (Travelling from Low to High)



Tracks Travelled = (95 – 30) + (95 – 5) + (25 – 5)

swap space management

Swap Space Management

- Virtual memory systems use the disk as an extension of main memory to support large logical address space
  - Frames are moved out to swap space on disks

Swap Space Management

- Implementation has some possibilities
  - A large file
    - Ordinary performance
  - Placing the swap-space in a separate partition and use special algorithms to allocate and de-allocate space
    - Very fast
    - Needs more work in configuration of the system when there is a change

raid

RAID

- Redundant Arrays of Independent Disks (RAID) is a means to achieve high reliability and performance with redundancy and parallelism
  - Applying redundancy to achieve a higher reliability
  - Applying parallelism to achieve a higher performance
- Connecting a large number of disks to a computer system in several specific manners

RAID

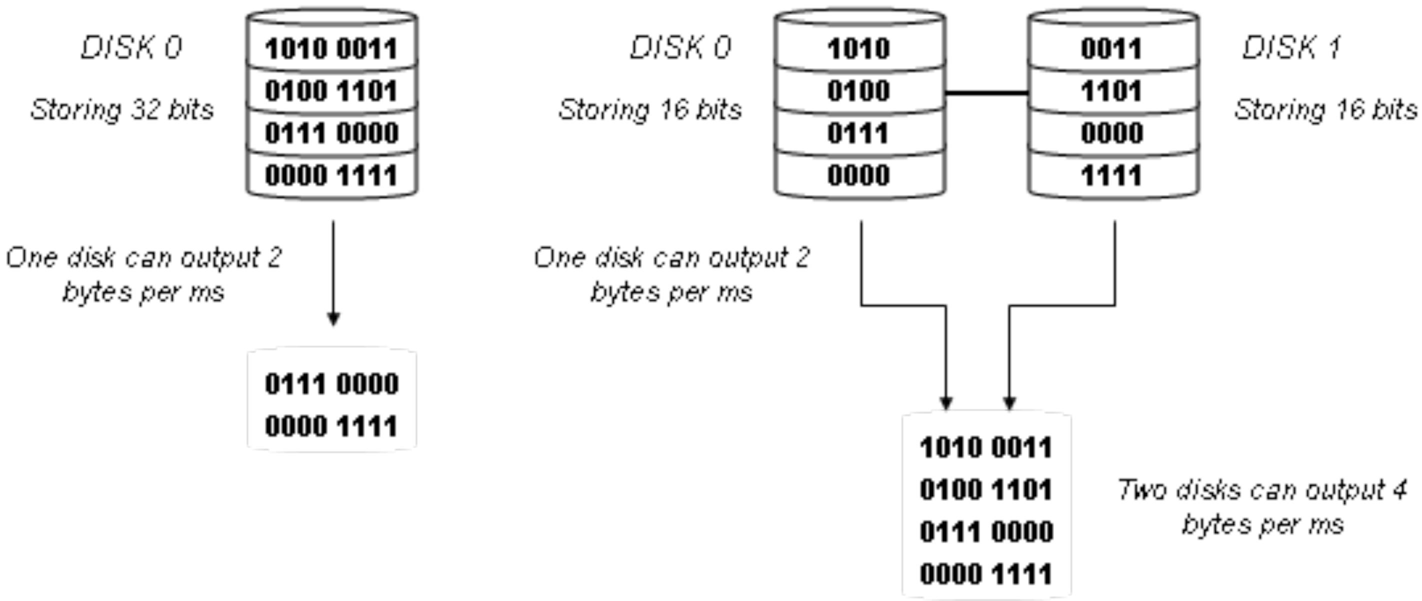
- Greater reliability through redundancy is to mirror or duplicate every disk
  - Expansive in terms of time and hardware cost
- Data striping is the approach to split the bits of each byte between several disks
  - The transfer rate is improved because a read can be carried out simultaneously on the disks
  - A recombination process the original bytes are recovered

RAID Levels

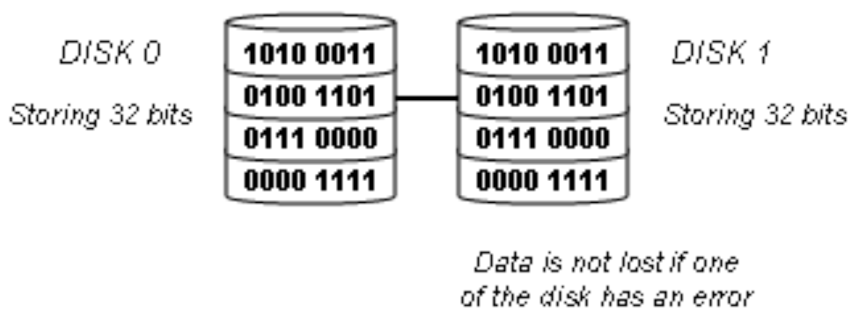
- RAID levels are specified ways to connect multiple hard disks together.
  - Each has its own characteristics in redundancy and performance.
- Common RAID levels include RAID 0, 1 and 5
  - RAID level 0 applies striping at block level and no redundancy
  - RAID level 1 mirrors the content of one disk on another disk
  - RAID level 5 is called block-interleaved distributed parity
    - Parity of data bytes and distributes the parity across all disks (except the disk where the data locates)
    - The parity bit allows a level of error correction

RAID Levels

RAID-0



RAID-1



RAID Levels

RAID-5