操作系统原理与设计

第12章 Mass-Storage structure（外存）

陈香兰

中国科学技术大学计算机学院

2009年09月01日

提纲

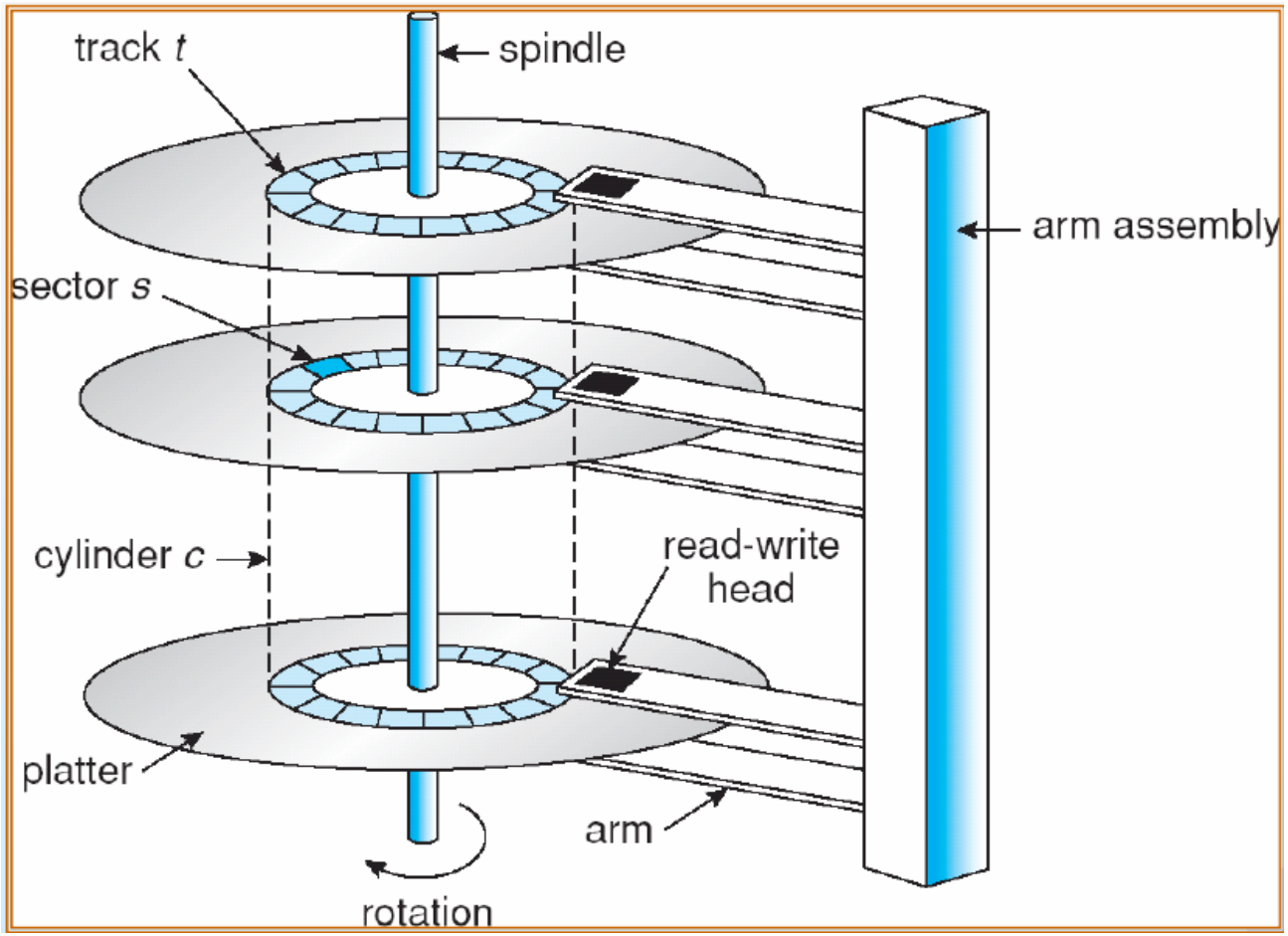
Overview of Mass Storage Structure I

- Magnetic disks provide bulk of secondary storage of modern computers
  - Drives rotate at 60 to 200 times per second
  - Transfer rate is rate at which data flow between drive and computer
  - Positioning time (random-access time) is time to move disk arm to desired cylinder (seek time) and time for desired sector to rotate under the disk head (rotational latency)
  - Head crash results from disk head making contact with the disk surface
    - That's bad
- Disks can be removable

Overview of Mass Storage Structure II

- Drive attached to computer via I/O bus
  - Busses vary, including EIDE, ATA, SATA, USB, Fibre Channel, SCSI
  - Host controller in computer uses bus to talk to disk controller built into drive or storage array

Overview of Mass Storage Structure III



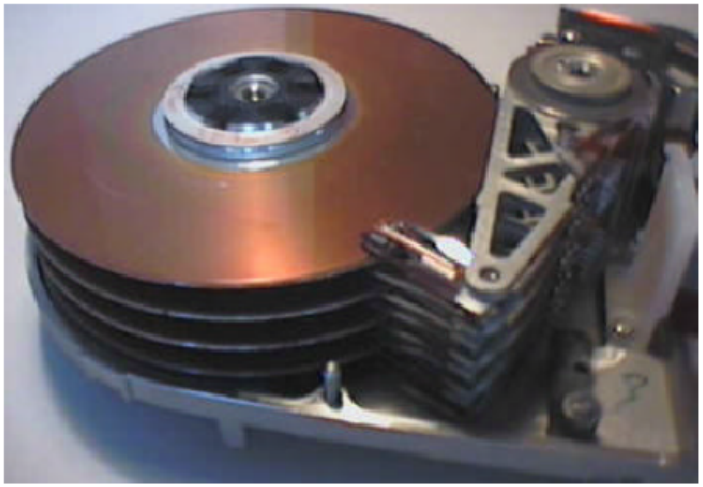
Overview of Mass Storage Structure (Cont.)

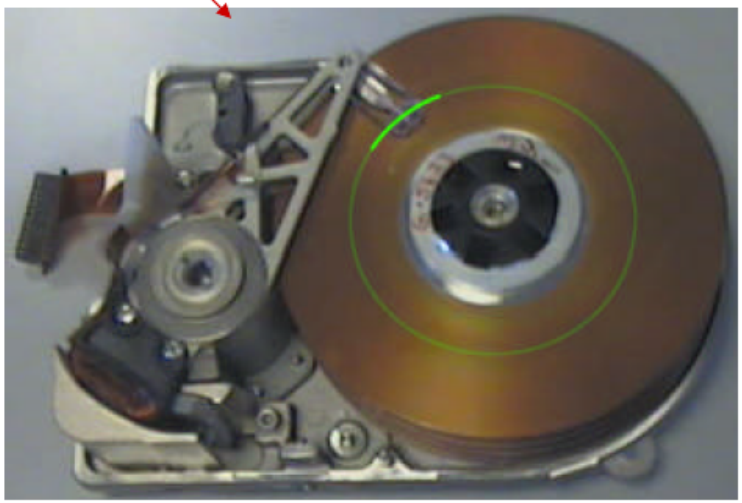
- Magnetic tape
  - Was early secondary-storage medium
  - Relatively permanent and holds large quantities of data
  - Access time slow
  - Random access ~1000 times slower than disk
  - Mainly used for backup, storage of infrequently-used data, transfer medium between systems
  - Kept in spool and wound or rewound past read-write head
  - Once data under head, transfer rates comparable to disk
  - 20-200GB typical storage
  - Common technologies are 4mm, 8mm, 19mm, LTO-2 and SDLT Oper

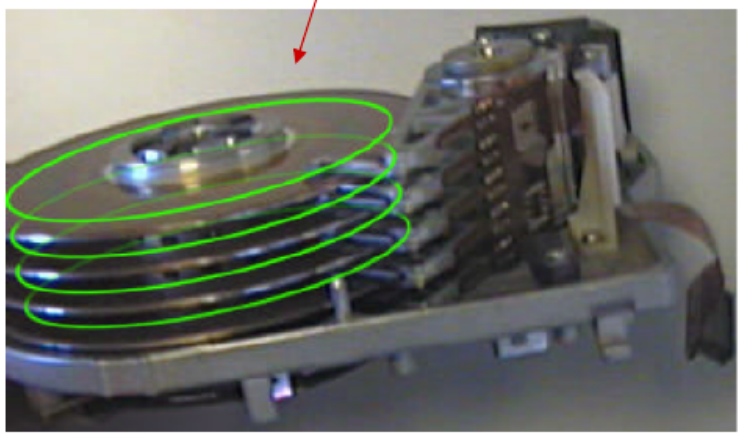
Disk Structure I

- Disk drives are addressed as large **1-D** arrays of logical blocks,
  - the **logical block** is the smallest unit of transfer.
  - usually, 512B
- The 1-D array of logical blocks is **mapped into the sectors** of the disk sequentially.
  - **Cylinder: track: sector**
  - **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 Structure II







- However in practise, the mapping is difficult because

Disk Scheduling I

- The OS is responsible for using hardware efficiently — for the disk drives,
  - this means having a fast access time and disk bandwidth.
- Access time has two major components
  1. **Seek time** is the time for the disk are to move the heads to the cylinder containing the desired sector.
  2. **Rotational latency** is the additional time waiting for the disk to rotate the desired sector to the disk head.
- **Minimize seek time**
- **Seek time $\approx$ seek distance**

Disk Scheduling II

- **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.

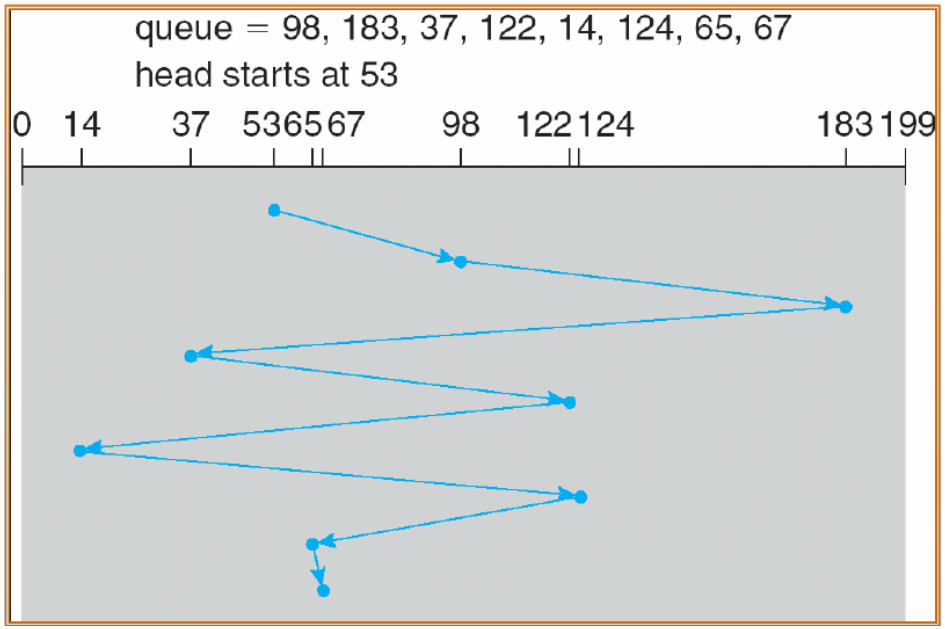
- **Request queue**
  - empty or not
- Several algorithms exist to schedule the servicing of disk I/O requests.
- We illustrate them with a request queue (**0-199**).

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

  Head points to **53** initially

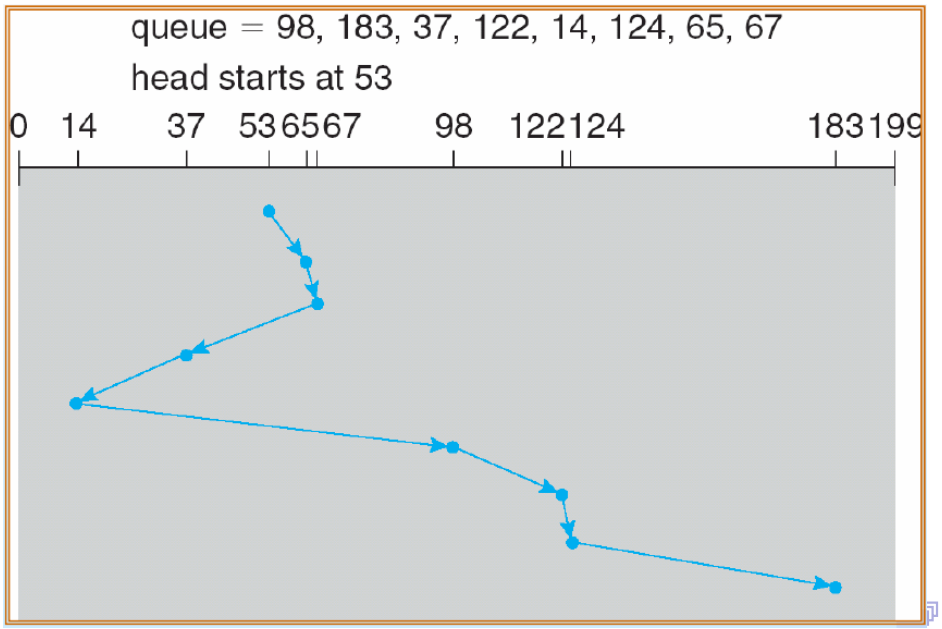
FCFS

- Simplest
- Total head movement $= \Sigma\left(h_i - h_{i-1}\right) =$
  $|98-53| + |183-98| + |37-183| + |122-37| +$
  $|14-122| + |124-14| + |65-124| + |67-65| = 640$



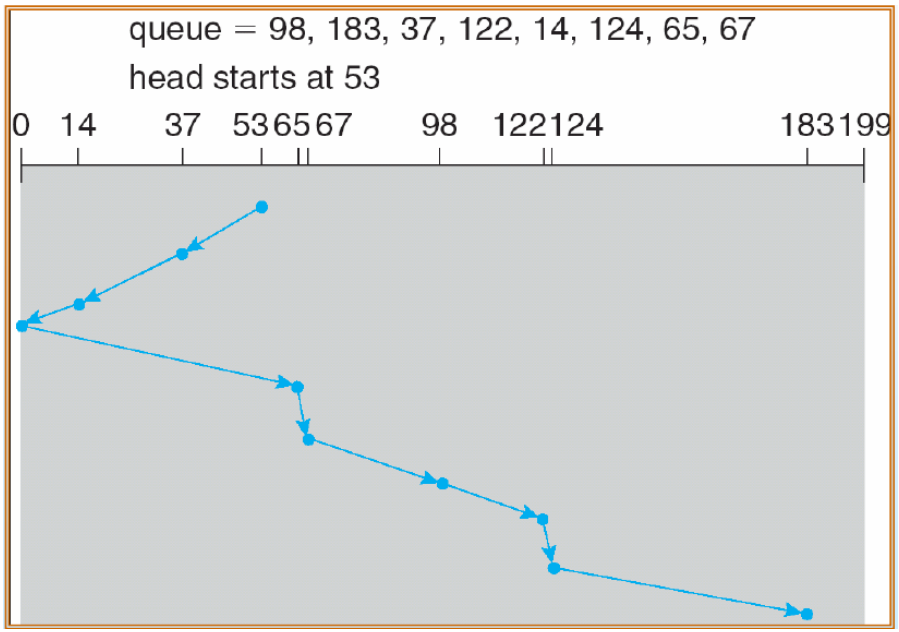
SSTF (shortest-seek-time-first)

- Selects the request with the **minimum seek time** from the current head position.
- SSTF≈SJF : **starvation**
- Total head movement= $|65-53|+|67-65|+|37-67|+|14-37|+|98-14|+|122-98|+|124-12|+|183-124|=236$
- Optimal?



SCAN (elevator algorithm)

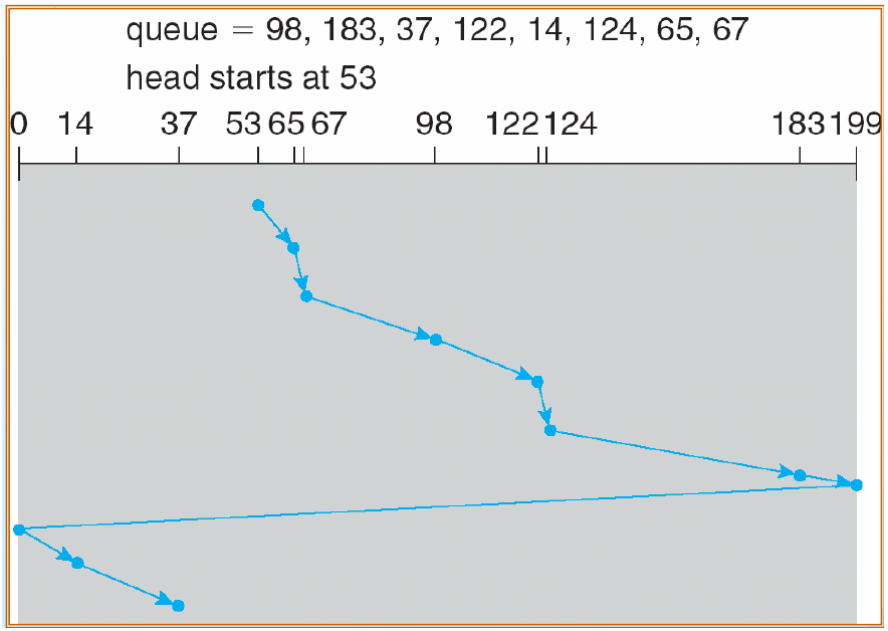
- The disk arm **starts at one end of the disk, and moves toward the other end**, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Illustration shows total head movement of 208 cylinders.
- Waiting time: Maximum is ?



C-SCAN I

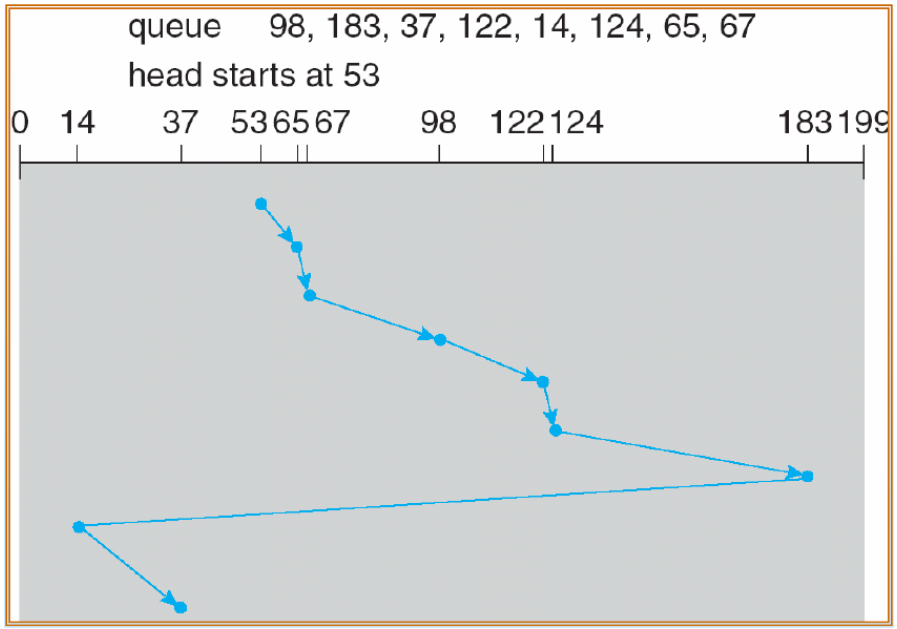
- Provides a more uniform wait time than SCAN.
- The head moves
  - from one end of the disk to the other.
  - servicing requests as it goes.
  - When it reaches the other end, however, it **immediately returns to the beginning of the disk**, without servicing any requests on the return trip.
- Treats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

C-SCAN II


queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
0
14
37
53
65
67
98
122
124
183
199

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.

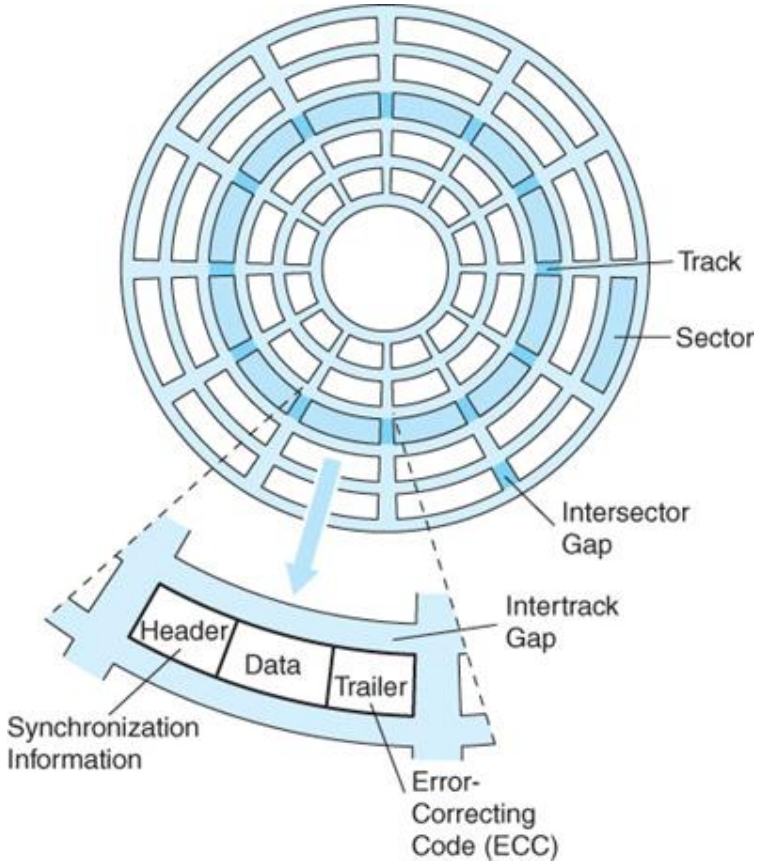


Selecting a Disk-Scheduling Algorithm

- 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, which can be influenced by
    - the file-allocation method
    - The location of directories and index blocks (caching?)
- The disk-scheduling algorithm should be written as a separate module of the OS, allowing it to be replaced with a different algorithm if necessary.
- Either SSTF or LOOK is a reasonable choice for the default algorithm.

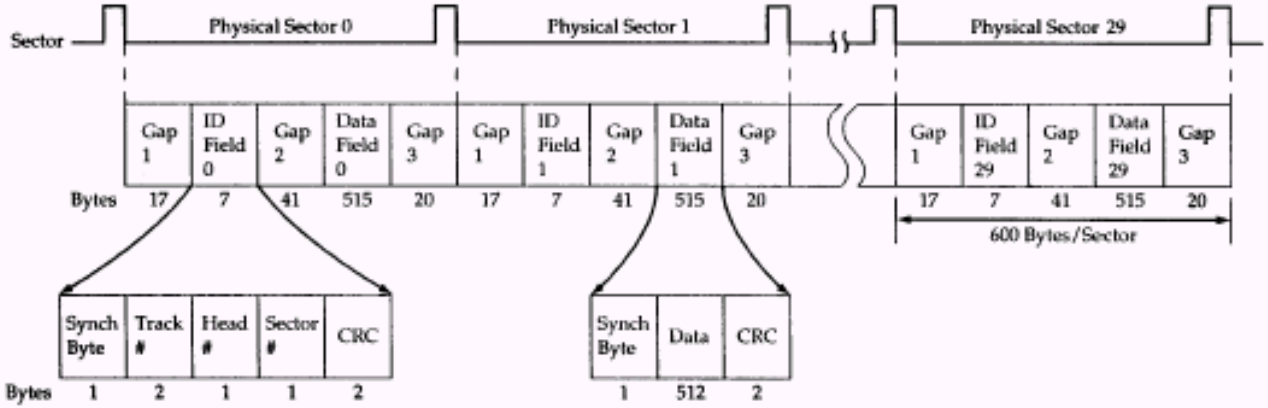
Disk Management I

- Disk Formatting
  - **Low-level formatting**, or **physical formatting**
    - Dividing a disk into sectors that the disk controller can read and write.



(From: http://tjliu.myweb.hinet.net/COA_CH_7.file

Disk Management II

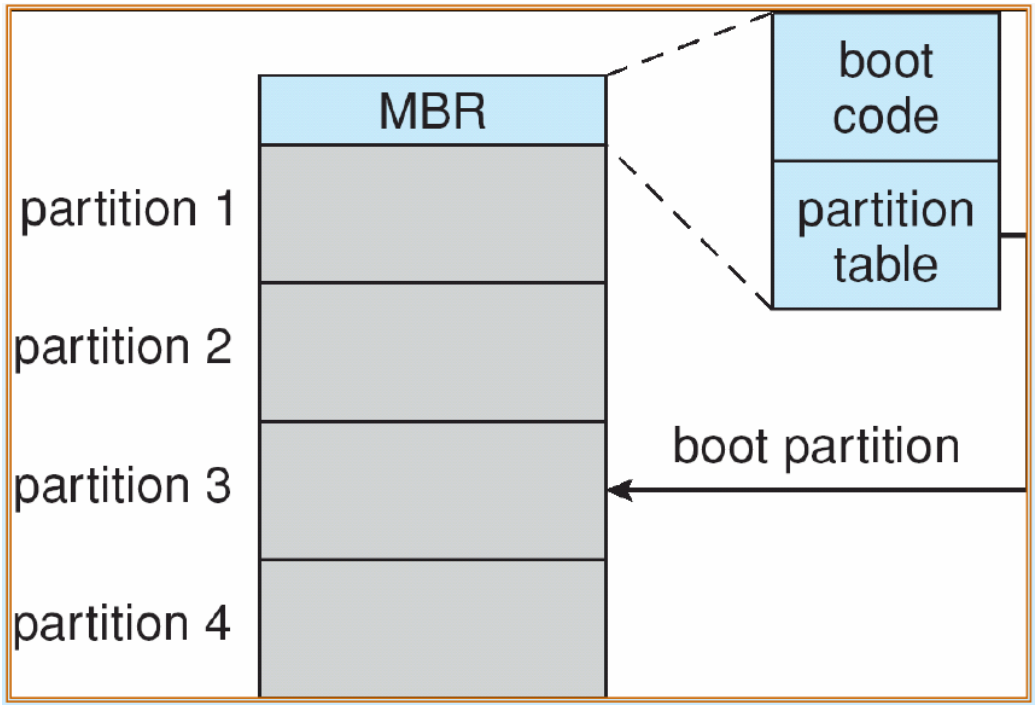


Disk Management III

- To use a disk to hold files, the operating system still needs to record its own data structures on the disk.
  - **Partition** the disk into one or more groups of cylinders.
    - **Logical formatting** or "making a file system".
- To increase efficiency, most FSes group blocks together into larger chunks, frequently called **clusters**
- **Boot block**
  - The (tiny) bootstrap is stored in ROM.
  - Mostly, the only job of bootstrap is to bring in a full bootstrap program from disk (boot disk, or system disk)
  - MBR
  - boot partition & boot sector

Disk Management IV

- Booting from a Disk in Windows 2000



Disk Management V

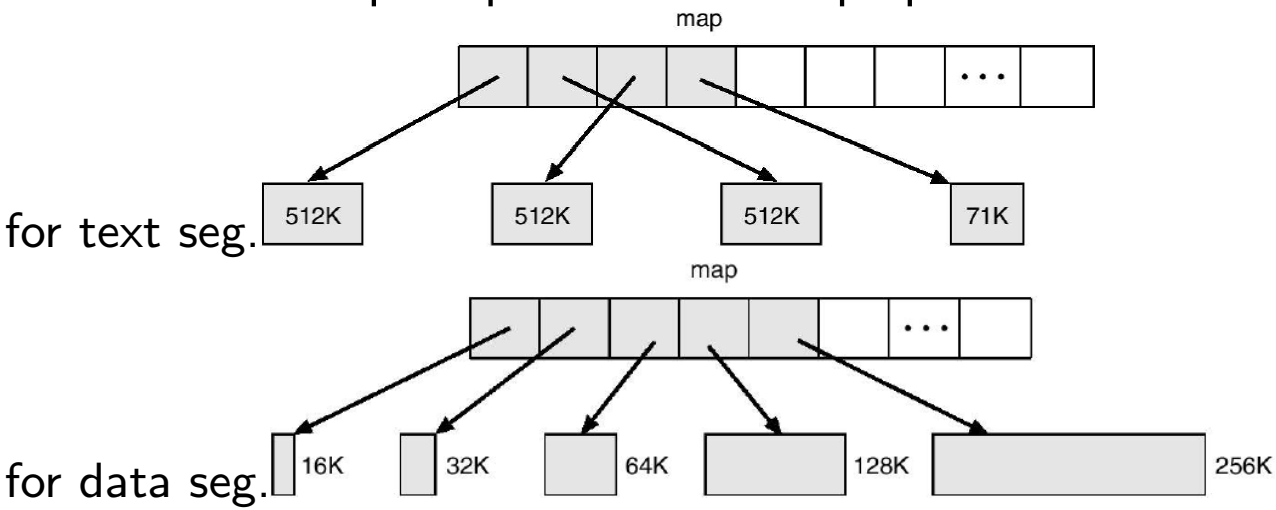
- Disk failure
  - complete failure VS. only one or more sectors become defective, **Bad blocks**
  - Methods towards bad blocks
    - manually: example, for MS-DOS, write a special value into FAT entry
    - sector sparing（备用）
    - sector slipping (滑动)

Swap-Space Management I

- Swapping & paging
  1. entire processes
  2. paging√
- **Swap-space**
  Virtual memory uses disk space as an extension of main memory.
- Swap-space can
  - be carved out of the normal file system
    - a large file with the file system
  - or, more commonly, it can be in a separate disk partition.

Swap-Space Management II

- Example1: 4.3BSD
  - allocates swap space when process starts;
  - holds text segment (the program) and data segment.
  - Kernel uses swap maps to track swap-space use.

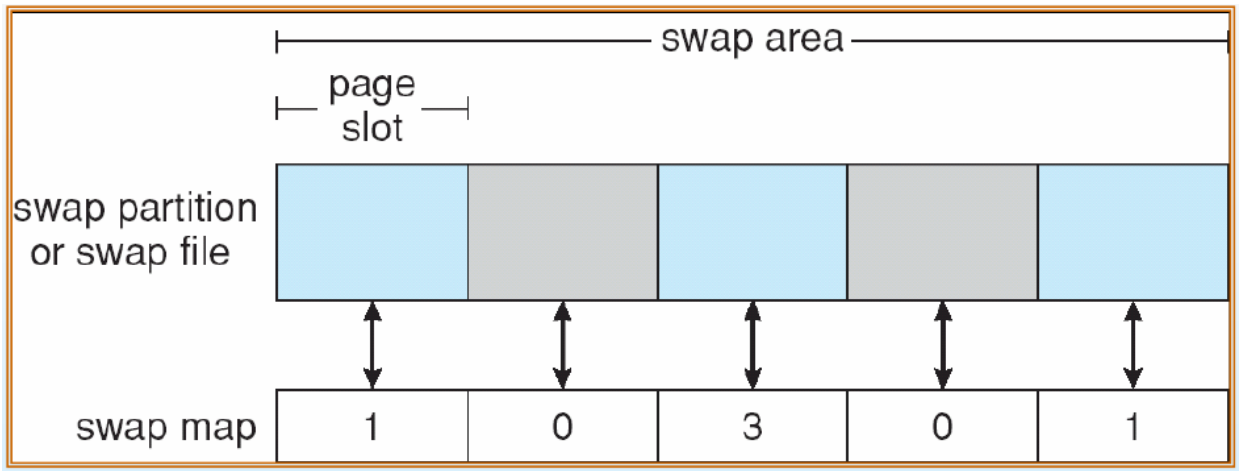


Swap-Space Management III

- Example2: Sorlaris
  - Version1: **for text segment**, no use of swap space; only used as a backing store for pages of anonymous memory, including memory allocated for stack, heap, uninitialized data
  - Version2: allocates swap space only when a page is forced out of physical memory, not when the virtual memory page is first created.

Swap-Space Management IV

- Example3: Linux
  - similar to Solaris1
  - allows one or more swap areas with 4KB slots
  - each swap area is associated with a swap map
    - 0: free; >0: occupied, sharing counts



小结

作业

- 华夏班：12.2
- 非华夏班：14.2

谢谢！