0117401: Operating System
计算机原理与设计

Chapter 12: Mass-Storage structure（外存）

陈香兰
xlanchen@ustc.edu.cn
http://staff.ustc.edu.cn/~xlanchen

Computer Application Laboratory, CS, USTC @ Hefei
Embedded System Laboratory, CS, USTC @ Suzhou

March 15, 2019

温馨提示：

为了您和他人的工作学习，
请在课堂上**关机或静音。**

不要在课堂上**接打电话。**

提纲

Overview of Mass Storage Structure

Disk Structure

Disk Scheduling (磁盘调度)

Disk Management

Swap-Space Management

RAID (磁盘阵列) Structure

小结和作业

Outline

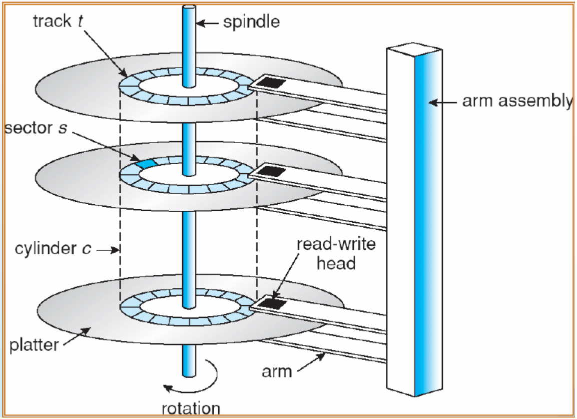
Overview of Mass Storage Structure

Overview of Mass Storage Structure

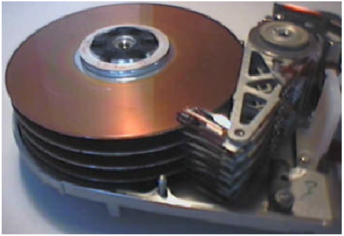
- **Magnetic disks (磁盘)** provide bulk of secondary storage of modern computers
  - Drives **rotate at 60 to 200 times per second**
  - **Transfer rate (传输速率)** is rate at which data flow between drive and computer
  - **Positioning time (random-access time)** is time to move disk arm to desired cylinder **(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

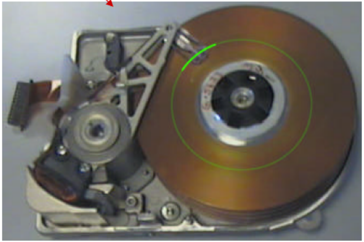
- Drive attached to computer via I/O bus
  - **Busses** vary, including EIDE, ATA, SATA, USB, Fibre Channel, SCSI
  - Host controller in computer uses bus to talk to disk controller built into drive or storage array



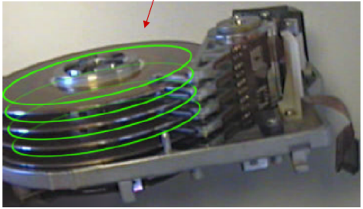
Overview of Mass Storage Structure



sector



cylinder



Overview of Mass Storage Structure

- **Magnetic tape (磁带)**
  - An **early** secondary-storage medium
  - Relatively permanent and holds **large** quantities of data
  - Access time **slow**
    - Random access $\sim$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

Outline

Disk Structure

- 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.
  - However, in practise, the mapping is difficult, because
    1. Defective sectors
    2. Sectors/track $\neq$ constant
       $\Rightarrow$ zones of cylinder

Outline

Disk Scheduling (磁盘调度)

Disk Scheduling (磁盘调度)

- 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 to move the heads to the cylinder containing the desired sector.
     - **Minimize seek time**
     - **Seek time $\approx$ seek distance**
  2. **Rotational latency** is the additional time waiting for the disk to rotate the desired sector to the disk head.
- **Disk bandwidth (磁盘带宽)** 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 (磁盘调度)

- **Request queue (请求队列)**
  - empty or not
- **How?**

  Several algorithms exist to schedule the servicing of disk I/O requests.

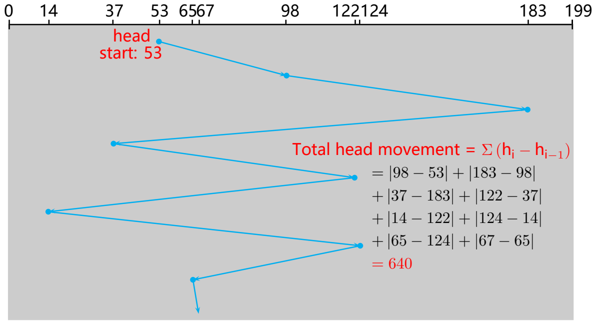
  1. FCFS
  2. SSTF (shortest-seek-time-first)
  3. SCAN (elevator algorithm)
  4. C-SCAN
  5. C-LOOK
- We illustrate them with a request queue (**0-199**).

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

  Head points to **53** initially

Disk Scheduling (磁盘调度)

1. **First Come, First Served (FCFS, 先来先服务)**
   - The simplest form of scheduling



Total head movement = $\Sigma\,(h_i - h_{i-1})$

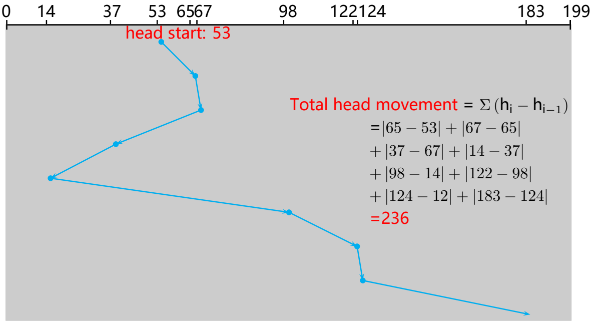
$$
\begin{aligned}
&= |98-53| + |183-98| \\
&+ |37-183| + |122-37| \\
&+ |14-122| + |124-14| \\
&+ |65-124| + |67-65| \\
&= 640
\end{aligned}
$$

request queue = 98, 183, 37, 122, 14, 124, 65, 67

Disk Scheduling (磁盘调度)

2. SSTF (shortest-seek-time-first)

- Selects the request with the **minimum seek time** from the current head position.



Total head movement $= \Sigma\,(h_i - h_{i-1})$

$$= |65 - 53| + |67 - 65| + |37 - 67| + |14 - 37| + |98 - 14| + |122 - 98| + |124 - 12| + |183 - 124|$$

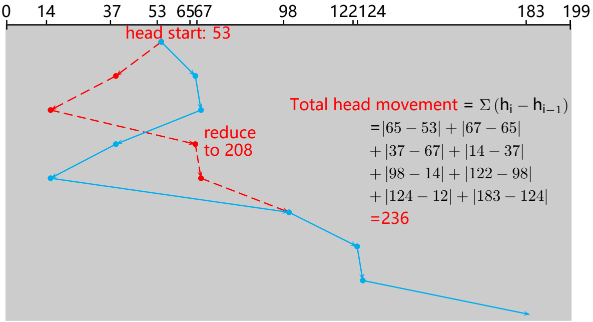
$=236$

request queue = 98, 183, 37, 122, 14, 124, 65, 67

- SSTF≈SJF : **starvation**

Disk Scheduling (磁盘调度)

2. SSTF (shortest-seek-time-first)
   - Selects the request with the **minimum seek time** from the current head position.



Total head movement $= \Sigma\,(h_i - h_{i-1})$

$$\begin{aligned} &= |65-53| + |67-65| \\ &+ |37-67| + |14-37| \\ &+ |98-14| + |122-98| \\ &+ |124-12| + |183-124| \\ &= 236 \end{aligned}$$

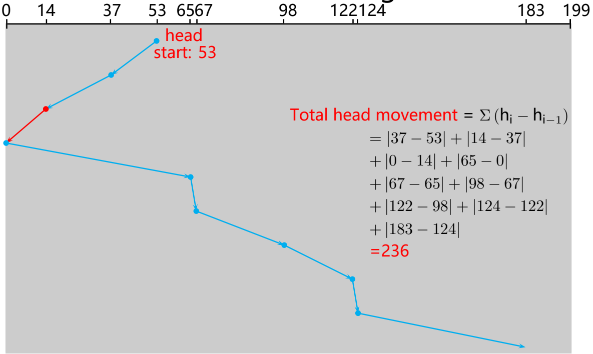
request queue = 98, 183, 37, 122, 14, 124, 65, 67

   - SSTF≈SJF : **starvation**
   - Optimal?

Disk Scheduling (磁盘调度)

3. **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.



Total head movement = $\Sigma\,(h_i - h_{i-1})$

$$= |37-53| + |14-37| + |0-14| + |65-0| + |67-65| + |98-67| + |122-98| + |124-122| + |183-124|$$

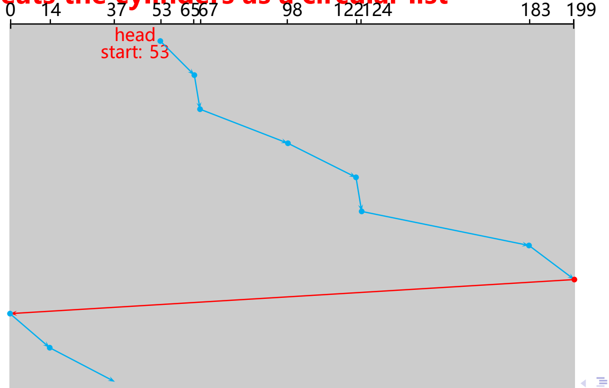
=236

request queue = 98, 183, 37, 122, 14, 124, 65, 67

- **Waiting time**: Maximum is ?

Disk Scheduling (磁盘调度)

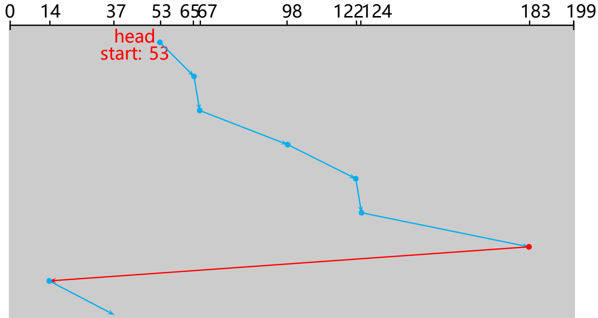
4. **C-SCAN**: **Provides a more uniform wait time** than SCAN.
   - The head **moves from one end of the disk to the other**, servicing requests as it goes. When it reaches the other end, however, it **immediately returns to the beginning of the disk**, without servicing any requests on the return trip.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



Disk Scheduling (磁盘调度)

5. 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.



request queue = 98, 183, 37, 122, 14, 124, 65, 67

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
  1. The file-allocation method
  2. The location of directories and index blocks (caching?)
- Either SSTF or LOOK is a reasonable choice for the default algorithm.
- 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.

Outline

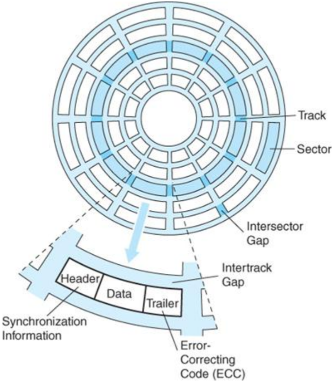
Disk Management

Disk Management

- Disk Formatting
- Boot Block
- Disk Failure

Disk Management

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

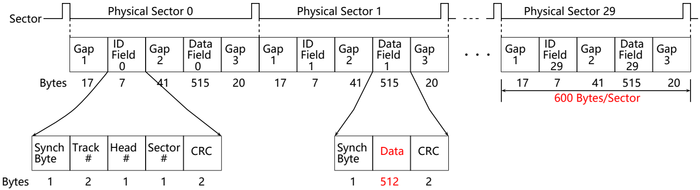


(From: http://tjliu.myweb.hinet.net/COA_CH_7.files/image055.jpg)

Disk Management

- Disk Formatting

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



Sector

Physical Sector 0 | Physical Sector 1 | . . . | Physical Sector 29

	Gap 1	ID Field 0	Gap 2	Data Field 0	Gap 3	Gap 1	ID Field 1	Gap 2	Data Field 1	Gap 3	. . .	Gap 1	ID Field 29	Gap 2	Data Field 29	Gap 3
Bytes	17	7	41	515	20	17	7	41	515	20		17	7	41	515	20

600 Bytes/Sector

	Synch Byte	Track #	Head #	Sector #	CRC
Bytes	1	2	1	1	2

Synch Byte	Data	CRC
1	512	2

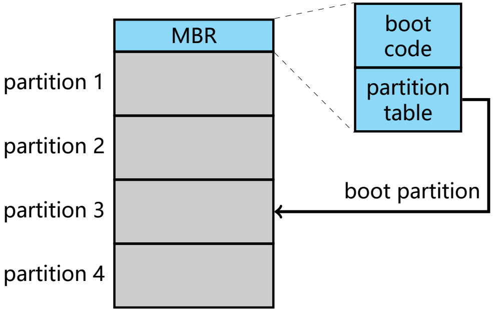
Disk Management

- Disk Formatting

2. To use a disk to hold files, the OS 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” .
3. To increase efficiency, most file-systems group blocks together into larger chunks, frequently called **clusters**

Disk Management

- **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)
  - Master boot record (MBR, 主引导记录)
  - Boot partition (启动分区) & boot sector (启动扇区)



Booting from a Disk in Windows 2000

Disk Management

- Disk failure
  - **Complete failure** VS. only one or more sectors become defective, **Bad blocks**
  - The data stored in bad blocks are lost.
  - **Methods** towards bad blocks
    1. **Manually**: example, for MS-DOS, write a special value into FAT entry
    2. **Sector sparing (备用)**
       (1) OS tries to read logical block 87;
       (2) The controller calculates the ECC and finds that sector is bad. It reports this finding to OS.
       (3) When rebooting, a special command is run to tell the SCSI controller to replace the bad sector with a spare;
       (4) After that, whenever logical block 87 is requested, the request is translated into the replacement sector' s address by the controller.
       Most disks are formatted to provide **a few spare sectors in each cylinder and a spare cylinder** as well.
    3. **Sector slipping (滑动)**

Disk Management

- Disk failure
  - **Complete failure** VS. only one or more sectors become defective, **Bad blocks**
  - The data stored in bad blocks are lost.
  - **Methods** towards bad blocks
    1. **Manually**: example, for MS-DOS, write a special value into FAT entry
    2. **Sector sparing (备用)**
    3. **Sector slipping (滑动)**

       Example:

       (1) Logical block 17 is bad

       (2) Logical blocks 18~202 are used, and 203 is available.

       (3) 202→203, 201→202, ..., 17→18

Outline

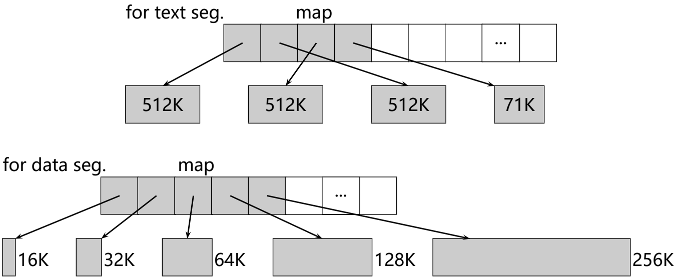
Swap-Space Management

Swap-Space Management

- Swapping & paging
  1. Entire processes
  2. Paging√
- **Swap-space** (对换空间)
  Virtual memory uses disk space as an extension of main memory.
  1. It can be carved out of the normal file system
     - **A large file** with the file system
  2. Or, more commonly, it can be in **a separate disk partition**.

Swap-Space Management

- **Example1: 4.3BSD**
  1. Allocates swap space when process starts;
  2. Holds text segment (the program) and data segment.
  3. Kernel uses swap maps to track swap-space use.

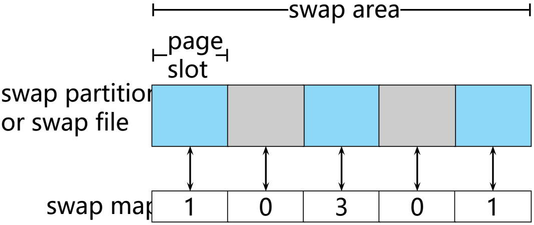


Swap-Space Management

- **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

- **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



Outline

RAID (磁盘阵列) Structure

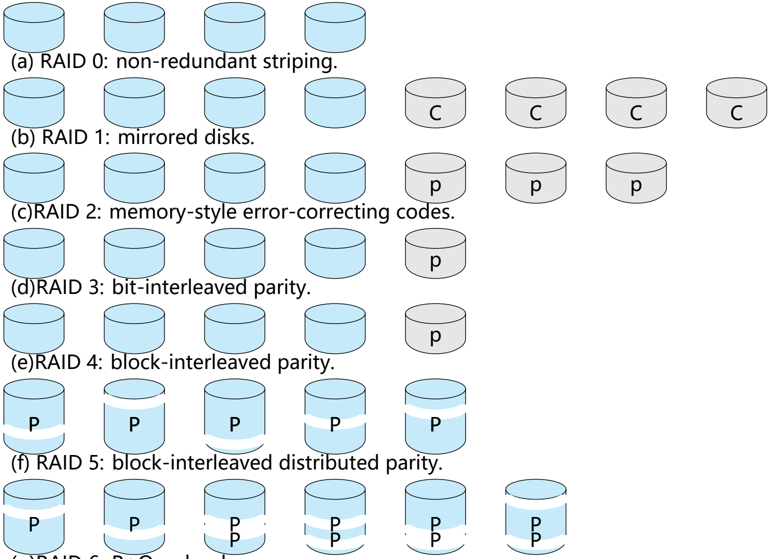
RAID (磁盘阵列) Structure

- **Redundant arrays of inexpensive disks (RAIDs, 磁盘阵列)** – Multiple disk drives provides
  - reliability via redundancy
  - higher data-transfer rate
- RAID is arranged into six different levels.

RAID (cont)

- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively.
- Disk striping uses a group of disks as one storage unit.
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data.
  - Mirroring or shadowing keeps duplicate of each disk.
  - Block interleaved parity uses much less redundancy.

RAID Levels



(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c)RAID 2: memory-style error-correcting codes.

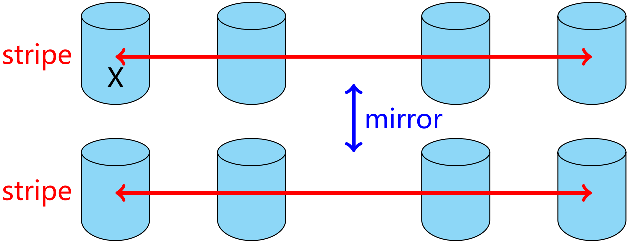
(d)RAID 3: bit-interleaved parity.

(e)RAID 4: block-interleaved parity.

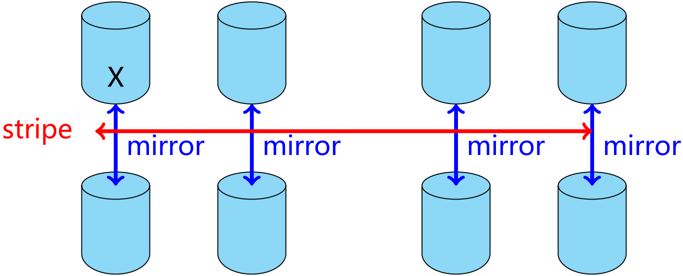
(f) RAID 5: block-interleaved distributed parity.

(g)RAID 6: P+Q redundancy.

RAID (0 + 1) and (1 + 0)



(a) RAID 0 + 1 with a single disk failure.



(b) RAID 1 + 0 with a single disk failure.

Outline

小结和作业

小结

Overview of Mass Storage Structure

Disk Structure

Disk Scheduling (磁盘调度)

Disk Management

Swap-Space Management

RAID (磁盘阵列) Structure

小结和作业

谢谢!