CS 3423 Operating Systems

Fall Semester 2019

Prof. Pai H. Chou

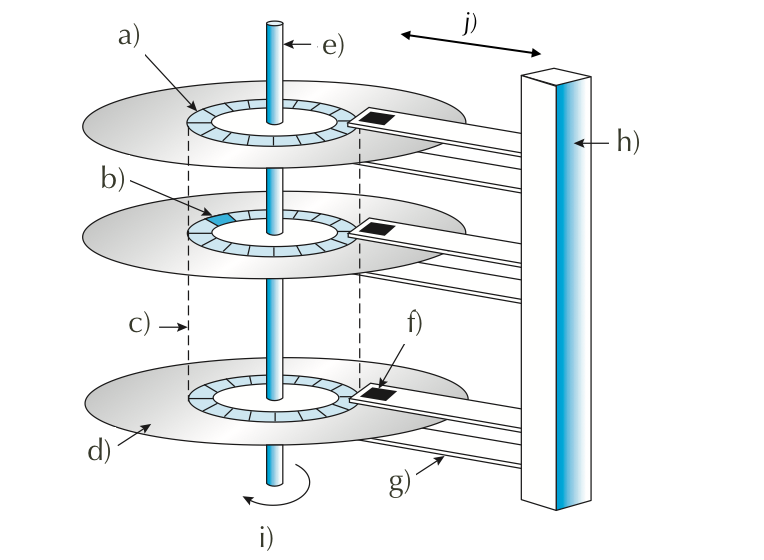
Assignment 12

Due Date: Sunday, December 8 2019, 11:59pm

Scope: Chapter 11, Mass-Storage Systems

## 1. Definitions and Short Answers

1. Name the following parts of a disk (a - h) and the motions (i - j).



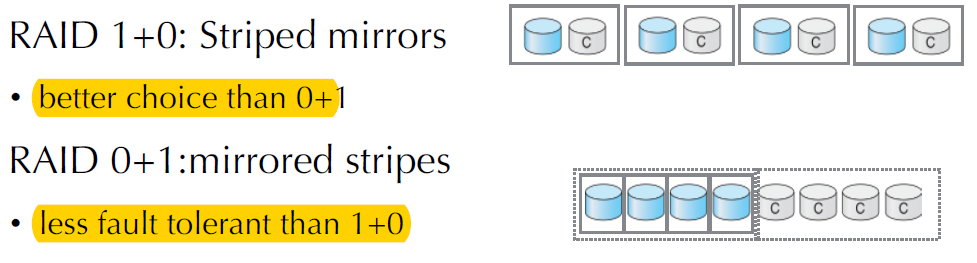
1. When accessing data on a magnetic disk,
   1. What is the **rotational latency**? time to rotate to desired sector
   2. What is the **seek time**? 前後移動找cylinder
   3. What is the **positioning time**?seek time + rotational latency
   4. What is another word for the positioning time? random access time
   5. What are the two components of positioning time in magnetic disk access?seek time + rotational latency
2. What is the difference between a cylinder and a track? cylinder = track的集合
3. For flash memory terminology, assume NAND flash,
   1. What is the minimum unit of reading? page
   2. What is the maximum unit of writing? page
   3. What is a block? minimum erase unit, contains several page
   4. What happens during an erase? set all bits to 1
   5. If you have to modify one byte, what are the steps involved? erase(all to 1) than write(1 to 0)
4. What is **wear-leveling**, and why is it important for flash memory? 關係到flash的壽命
5. What is the primary action that an OS can schedule to improve the performance of a hard disk drive? minimize seek time
6. Of the different disk scheduling algorithms,
   1. is FIFO in general a good policy for HDD? NO for SSD? YES
   2. is STSF in general a good policy for HDD? NO What kind of problem does it have? may cause starvation What about for SSD?no rotation no problem
   3. What is the difference between SCAN and C-SCAN? SCAN: 走到底轉彎；C-SCAN: 走到底回到出發點再走一次
   4. What is the difference between SCAN and LOOK? SCAN: 走到底；LOOK: 走到最邊的request
   5. Why does SCAN have more predictable behavior than LOOK? LOOK: 走到最邊的request
7. If NVM scheduling does not need to consider rotation or seek time, what does it need to consider? wear level, need to avoid write-amplification
8. What is a **spare** sector? 備用的sector
9. What is **sector-slipping**? 將資料移到備用的sector上
10. Can a regular file system be used for swap space? yes What are the advantages and disadvantages? disadv: 會很慢 ，adv: 不用去切空間，不用fixed sized
11. What is a **swap partition**, and why is it a good idea? swap partition: raw space, 切出來的，overhead較少
12. how do you pronounce SCSI?去死吧
13. What does **RAID** stand for? Redundant Array of Independent Disks
14. How can RAID achieve higher reliability? Redundancy
15. How can RAID achieve higher disk performance? striping In what metric?

Bit-level striping: access different bits on different disks

Block-level striping(common): access different blocks on different disks

1. What is the meaning of the following about disks?
   1. **mean time to failure** disk多久會掛掉
   2. **mean time to repair**, and is it related to mean time to failure?disk掛掉多久可以修好，
   3. **mean time to data loss** disk多久會丟失資料
2. What is the meaning of
   1. **mirroring**? Make exact copy of disk
   2. **data striping**? split data onto multiple disks, access in parallel Is it related to mirroring?
   3. What is the difference between **bit-level** and **block-level** striping? which is more common?Bit-level striping: access different bits on different disks

Block-level striping(common): access different blocks on different disks

* 1. What is the meaning of **striped mirror**? **mirrored stripes**? Which one is a better choice?

## 2. Programming Exercise

## 3. Disk Scheduling Algorithms [25 points]

You are to implement the disk (seek) scheduling algorithms covered in Chapter 11.

Use the following template ([download](https://drive.google.com/file/d/1R5EUj-fHPWaGPdlTFmtO1Jwyn-bOTnJc/view?usp=sharing) and rename as disk.py):

class DiskScheduler:

\_POLICIES = ['FCFS', 'SSTF', 'SCAN', 'LOOK', 'C-SCAN', 'C-LOOK']

def \_\_init\_\_(self, nCylinders):

self.nCylinders = nCylinders

def schedule(self, initPos, requestQueue, policy, direction):

'''

request is the list of cylinders to access

policy is one of the strings in \_POLICIES.

direction is 'up' or 'down' and applies to (C-)SCAN/LOOK only.

returns the list for the order of cylinders to access.

'''

if policy == 'FCFS':

# return the disk schedule for FCFS

if policy == 'SSTF':

# compute and return the schedule for shortest seek time first

if policy in ['SCAN', 'C-SCAN', 'LOOK', 'C-LOOK']:

# sequentially one direction to one end (up or down),

# then sequentially in opposite direction.

# compute and return the schedule accordingly.

def totalSeeks(initPos, queue):

lastPos = initPos

totalMoves = 0

for p in queue:

totalMoves += abs(p - lastPos)

lastPos = p

return totalMoves

if \_\_name\_\_ == '\_\_main\_\_':

def TestPolicy(scheduler, initHeadPos, requestQ, policy, direction):

s = scheduler.schedule(initHeadPos, requestQ, policy, direction)

t = totalSeeks(initHeadPos, s)

print('policy %s %s (%d): %s' % (policy, direction, t, s))

scheduler = DiskScheduler(200)

requestQueue = [98, 183, 37, 122, 14, 124, 65, 67]

initHeadPos = 53

for policy **in** DiskScheduler.\_POLICIES:

if policy[:2] == 'C-' or policy[-4:] in ['SCAN', 'LOOK']:

TestPolicy(scheduler,initHeadPos, requestQueue, policy, 'up')

TestPolicy(scheduler,initHeadPos, requestQueue, policy, 'down')

else:

TestPolicy(scheduler, initHeadPos, requestQueue, policy, '')

print('more tests on SCAN and C-SCAN')

rQs = [[98, 37, 0, 122, 14], [98, 37, 199, 122, 14], [98,0,37,199,14]]

for q **in** rQs:

print('Q=%s' % q)

for policy in ['SCAN', 'C-SCAN']:

for direction in ['up', 'down']:

TestPolicy(scheduler, initHeadPos, q, policy, direction)

You can expect to get output like this:

$ python3 disk.py

policy FCFS (640): [98, 183, 37, 122, 14, 124, 65, 67]

policy SSTF (236): [65, 67, 37, 14, 98, 122, 124, 183]

policy SCAN up (331): [65, 67, 98, 122, 124, 183, 199, 37, 14]

policy SCAN down (236): [37, 14, 0, 65, 67, 98, 122, 124, 183]

policy LOOK up (299): [65, 67, 98, 122, 124, 183, 37, 14]

policy LOOK down (208): [37, 14, 65, 67, 98, 122, 124, 183]

policy C-SCAN up (382): [65, 67, 98, 122, 124, 183, 199, 0, 14, 37]

policy C-SCAN down (386): [37, 14, 0, 199, 183, 124, 122, 98, 67, 65]

policy C-LOOK up (322): [65, 67, 98, 122, 124, 183, 14, 37]

policy C-LOOK down (326): [37, 14, 183, 124, 122, 98, 67, 65]

more tests on SCAN and C-SCAN

Q=[98, 37, 0, 122, 14]

policy SCAN up (345): [98, 122, 199, 37, 14, 0]

policy SCAN down (175): [37, 14, 0, 98, 122]

policy C-SCAN up (382): [98, 122, 199, 0, 14, 37]

policy C-SCAN down (353): [37, 14, 0, 199, 122, 98]

Q=[98, 37, 199, 122, 14]

policy SCAN up (331): [98, 122, 199, 37, 14]

policy SCAN down (252): [37, 14, 0, 98, 122, 199]

policy C-SCAN up (382): [98, 122, 199, 0, 14, 37]

policy C-SCAN down (353): [37, 14, 0, 199, 122, 98]

Q=[98, 0, 37, 199, 14]

policy SCAN up (345): [98, 199, 37, 14, 0]

policy SCAN down (252): [37, 14, 0, 98, 199]

policy C-SCAN up (382): [98, 199, 0, 14, 37]

policy C-SCAN down (353): [37, 14, 0, 199, 98]