PERSISTENCE: DISK SCHEDULING

Shivaram Venkataraman

CS 537, Spring 2020

ADMINISTRIVIA

Project 4a is out! Due April 2th → Piazza

More details in discussion section

Midterm grading in progress

Audio quality ?

Chat

Me

Yien

AGENDA / LEARNING OUTCOMES

How do you calculate sequential and random tput of a disk?

What algorithms are used to schedule I/O requests?

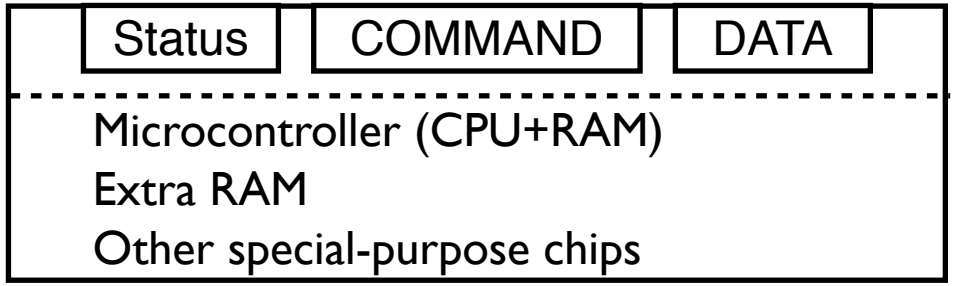
RECAP

Virtualization
Concurrency
Persistence

EXAMPLE WRITE PROTOCOL

Example
xv6 → IDE.c driver

General / Simple




```
while (STATUS == BUSY)
   ; // spin
Write data to DATA register
Write command to COMMAND register
while (STATUS == BUSY)
   ; // spin
```

Interrupts

DMA

Interrupts

Device drivers
↳ Software / code that interacts with the device

Memory → Device

IN/OUT
MMIO

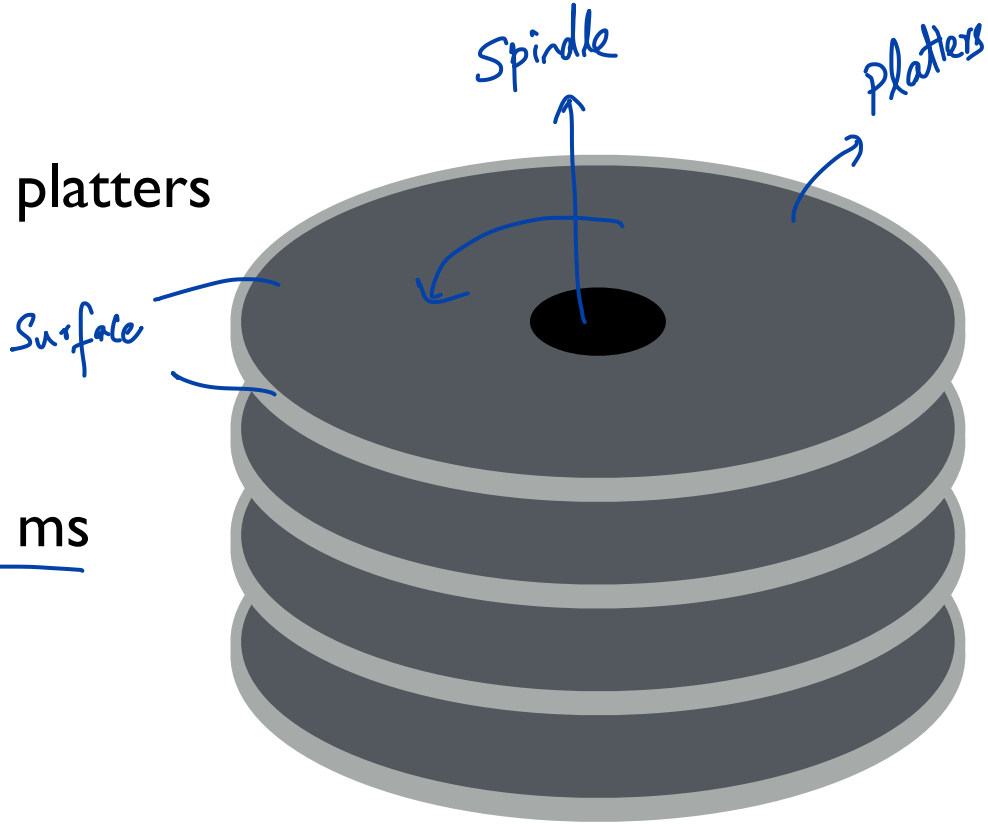
RPM

Hard disk drives

Motor connected to spindle spins platters

Rate of rotation: RPM

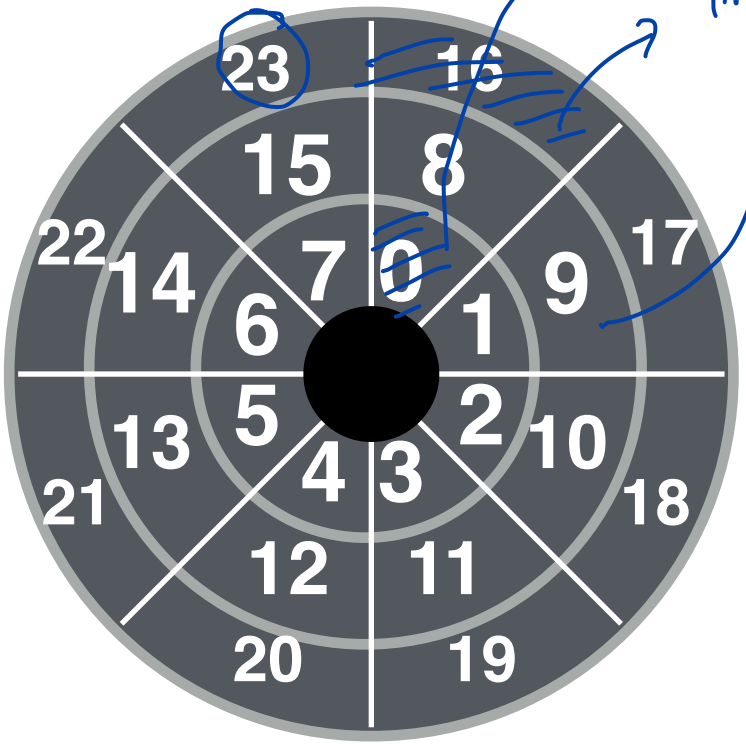
10000 RPM → single rotation is 6 ms



read / write to a sectors

Tracks are divided into numbered sectors → 512 bytes

Analogy Pages

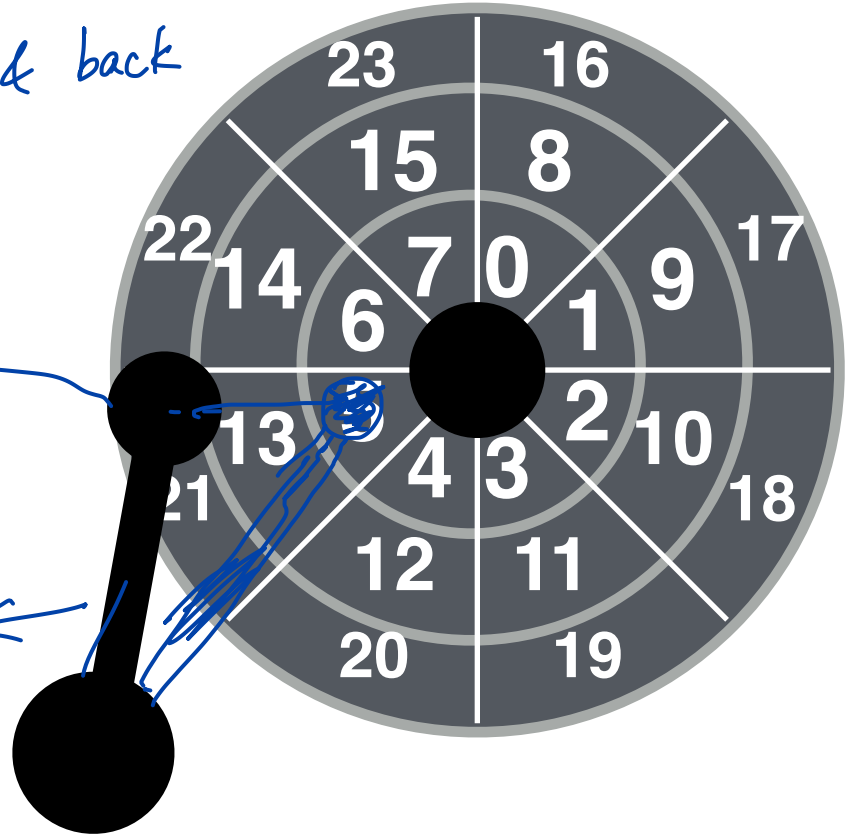


Arm can move from outer track to inner track & back

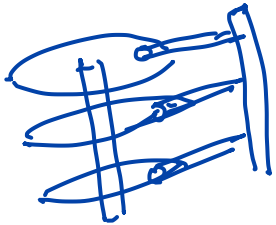
Heads on a moving arm can read from each surface.

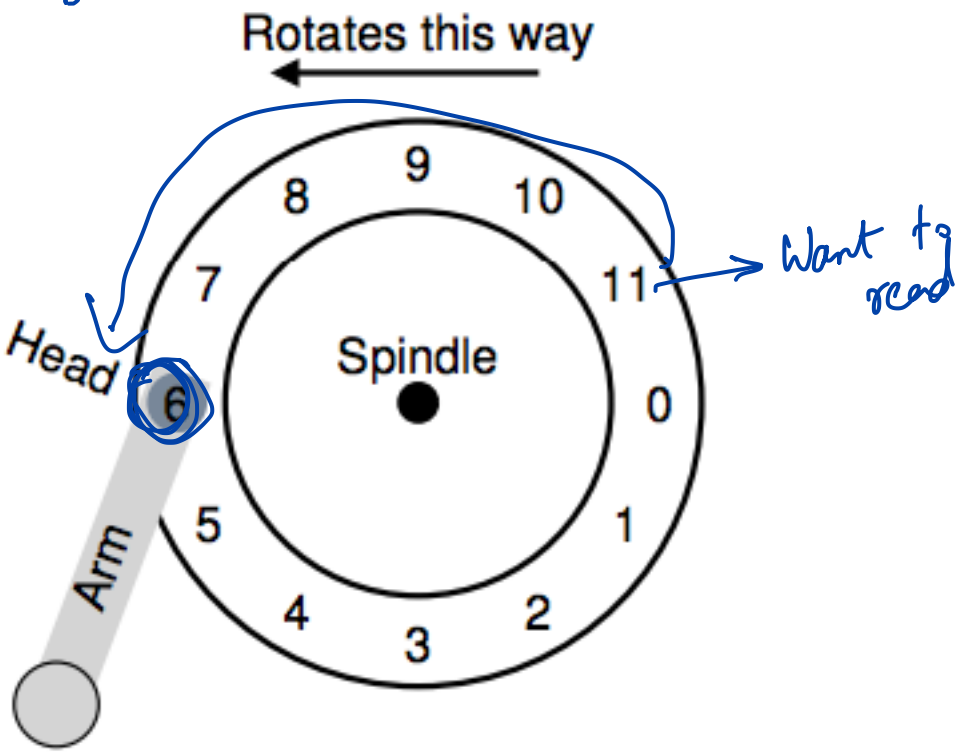
Does the read/write

Head



READING DATA FROM DISK

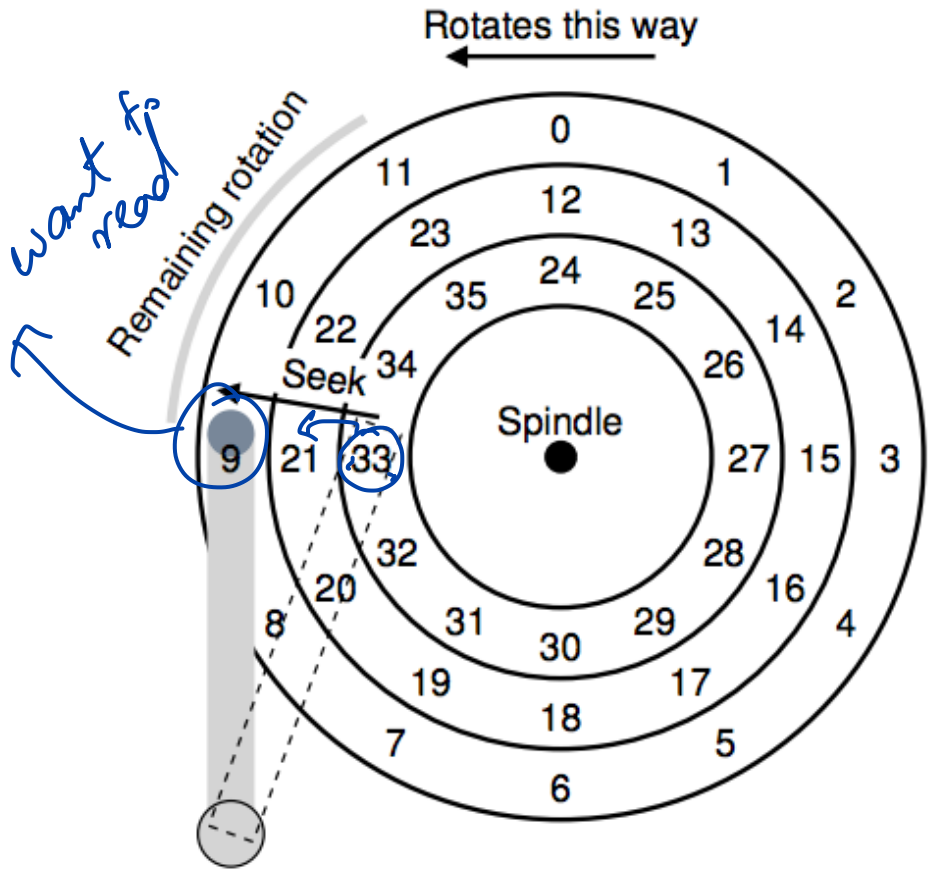




Rotational delay

- Wait for disk to rotate
- Once sector is under the Head do the read/write

READING DATA FROM DISK



Seek Time

Time it takes for the arm to move to the right track

TIME TO READ/WRITE

Three components:

Time = seek + rotation + transfer time

to get to right track

right sector inside track

You want to perform the operation

SEEK, ROTATE, TRANSFER

Seek cost: Function of cylinder distance

Not purely linear cost

Must accelerate, coast, decelerate, settle

Settling alone can take 0.5 - 2 ms

stopping

Entire seeks often takes 4 - 10 ms

Average seek = 1/3 of max seek

Derivation in text book

Depends on rotations per minute (RPM)

7200 RPM is common, 15000 RPM is high end

Average rotation?

Half of time for 1 rotation

Pretty fast: depends on RPM and sector density.

100+ MB/s is typical for maximum transfer rate

Bus speed

USB 2.0 slower

QUIZ 21

https://tinyurl.com/cs537-sp20-quiz21

What is the time for 4KB random read with Cheetah?

6 ms

	Cheetah 15K.5	Barracuda
Capacity	300 GB	1 TB
RPM	15,000	7,200
Average Seek	4 ms	9 ms
Max Transfer	125 MB/s	105 MB/s
Platters	4	4
Cache	16 MB	16/32 MB
Connects via	SCSI	SATA

$= T_{seek} + T_{rotation} + T_{transfer}$

$= 4\,ms + 2\,ms + 0.03 = \boxed{6\,ms}$

15000 Rotations in 60,000 ms

1 rotation in 4 ms

Avg $= \frac{4}{2} = 2\,ms$

$T_{transfer} = \frac{4\,KB}{125\,MB/s} = 0.032\,ms$

QUIZ 21

https://tinyurl.com/cs537-sp20-quiz21

13.19 ms

What is the time for 4KB random read with Barracuda?

	Cheetah 15K.5	Barracuda
Capacity	300 GB	1 TB
RPM → 60s = 60,000 ms	15,000	7,200
Average Seek	4 ms	9 ms
Max Transfer	125 MB/s	105 MB/s
Platters	4	4
Cache	16 MB	16/32 MB
Connects via	SCSI	SATA

$T_{seek} + T_{rot} + T_{transfer}$

$9ms + 4.16ms + 0.038$

7200 rot in 60000 ms

1 rot $\frac{60000}{7200}$ ms = 8.33 ms

Avg rot = $\frac{8.33}{2}$ = 4.16 ms

$T_{transfer} = \frac{4\ KB}{105\ MB/s} = 0.038ms$

WORKLOAD PERFORMANCE

WORKLOAD PERFORMANCE

So…

- seeks are slow → ≃ 4-9ms
- rotations are slow → ≃ 2-4ms
- transfers are fast → 0.03 ms

Avoid seeks and rotations

How does the kind of workload affect performance?

Sequential: access sectors in order → Better

Random: access sectors arbitrarily → Worse

DISK SPEC

	Cheetah	Barracuda
Capacity	300 GB	1 TB
RPM	15,000	7,200
Avg Seek	4 ms	9 ms
Max Transfer	125 MB/s	105 MB/s
Platters	4	4
Cache	16 MB	32 MB

Transfer time dominates

Sequential workload: what is throughput for each?

100 MB data sequentially

6ms to seek + rotate

$\simeq \frac{100\,MB}{125\,MB/s} \simeq 1s$

OTHER IMPROVEMENTS

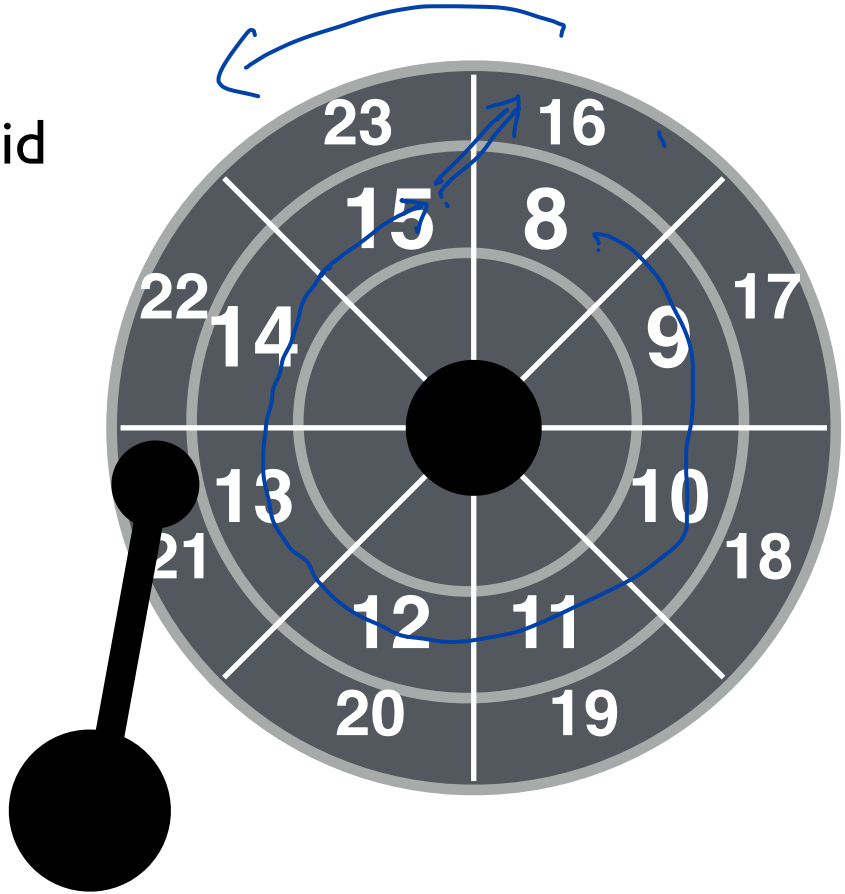
Track Skew

Zones

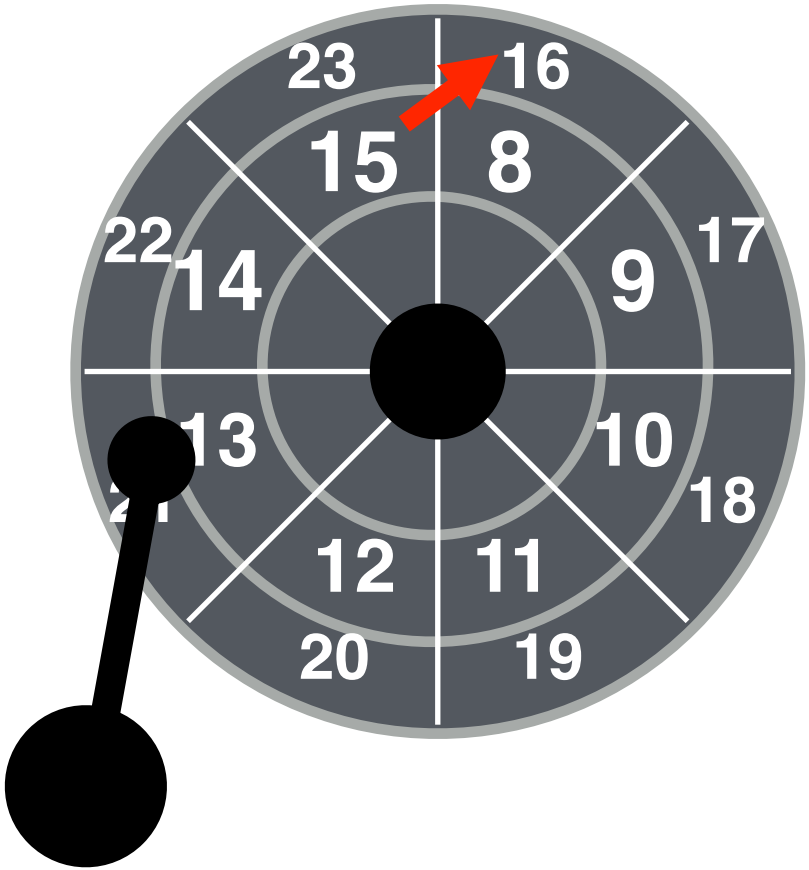
Cache

Imagine sequential reading,
how should sectors numbers be laid out on disk?

Rotation is always happening and is simultaneous with arm movement



When reading 16 after 15, the head won't settle quick enough, so we need to do a rotation.

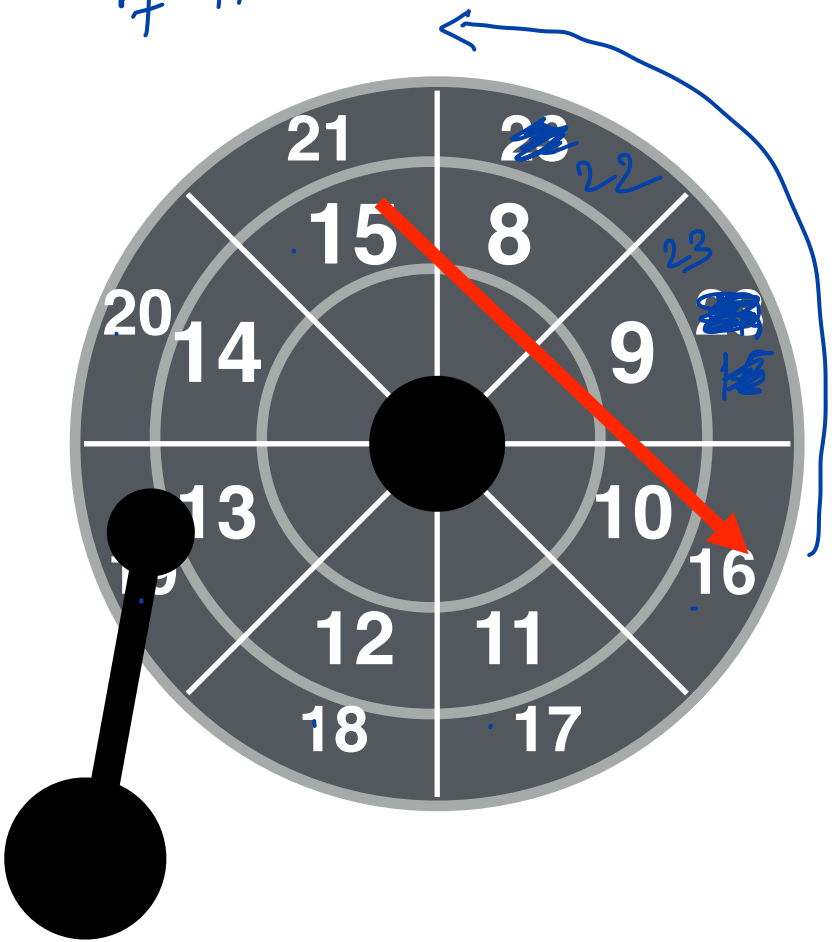


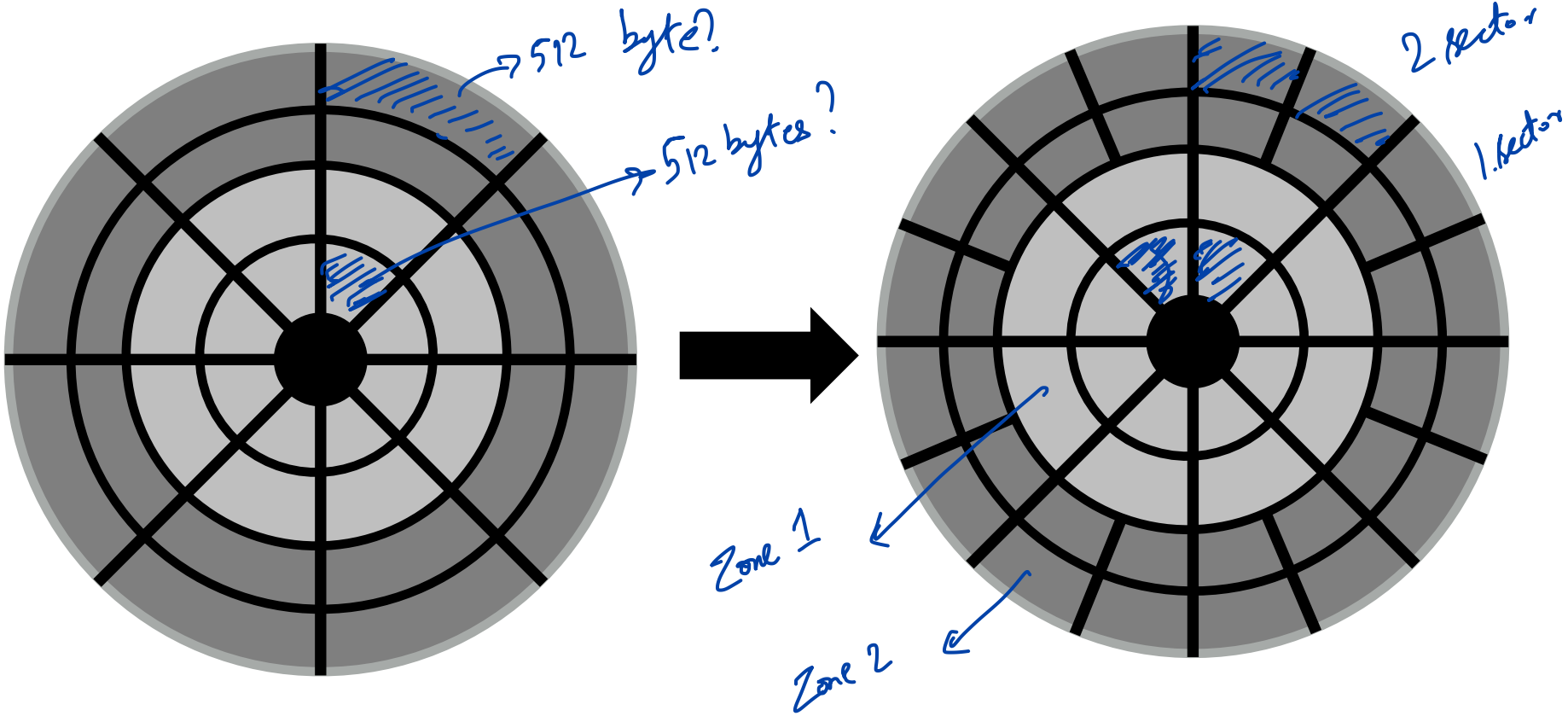
Random (4KB) = seek + seq read of 4KB
↓
sequential

Track Skew

↓

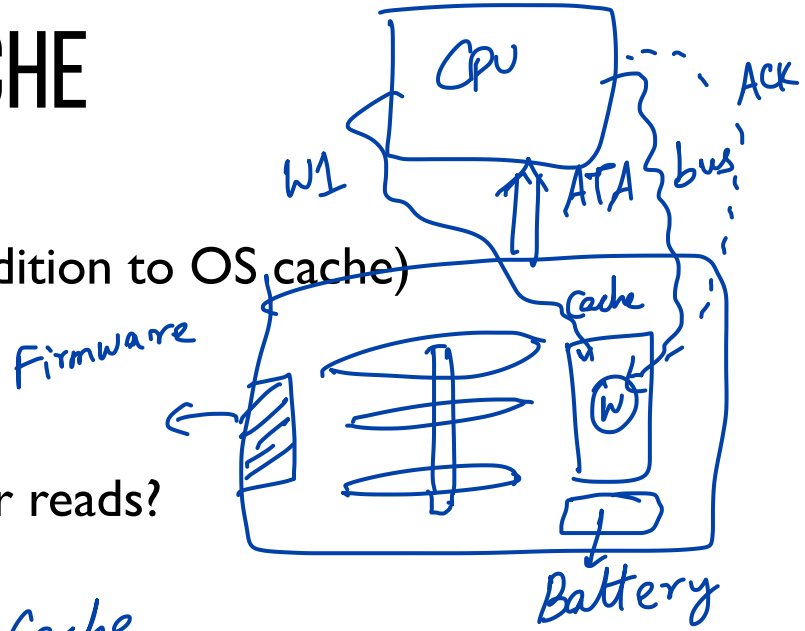
Account for rotation & seek happening at the same time





ZBR (Zoned bit recording): More sectors on outer tracks

DRIVE CACHE



Drives may cache both reads and writes. (In addition to OS cache)

What advantage does caching in **drive** have for reads?

Avoid doing disk seek if sector is in Cache

What advantage does caching in **drive** have for writes?

Acknowledge writes before it is "persisted"

Power failure then data loss??

BUFFERING

Disks contain internal memory (2MB-16MB) used as cache

Read-ahead: "Track buffer"

- Read contents of entire track into memory during rotational delay

Write caching with volatile memory

- Immediate reporting: Claim written to disk when not
- Data could be lost on power failure

Tagged command queueing

- Have multiple outstanding requests to the disk
- Disk can reorder (schedule) requests for better performance

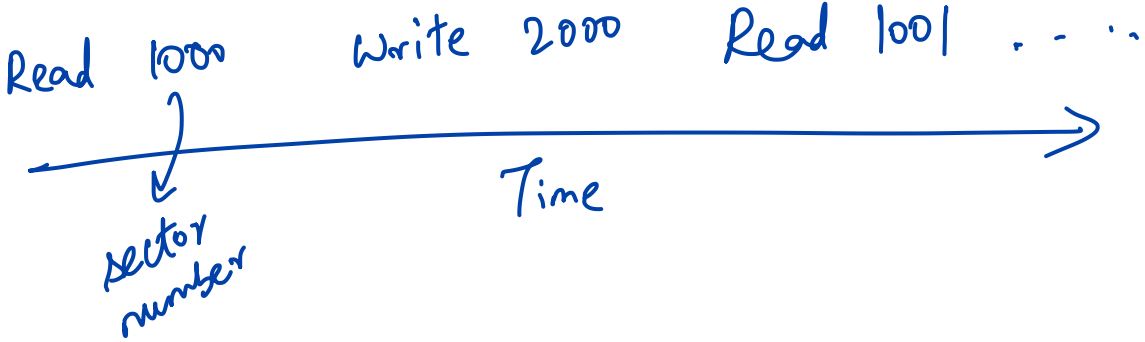
I/O SCHEDULERS

I/O SCHEDULERS

Given a stream of I/O requests, in what order should they be served?

Much different than CPU scheduling

Position of disk head relative to request position matters more than length of job



FCFS (FIRST-COME-FIRST-SERVE) → FIFO

Assume seek+rotate = 10 ms for random request

How long (roughly) does the below workload take?Requests are given in sector numbers

300001, 700001, 300002, 700002, 300003, 700003 ≃ 60 ms

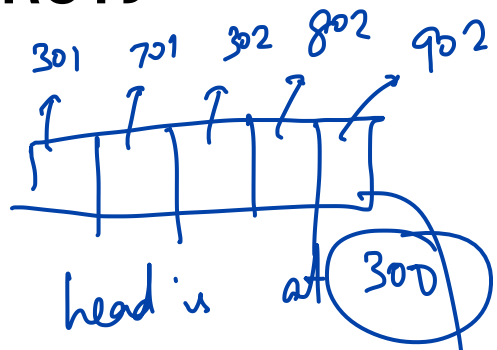
10ms 10ms 10ms

300001, 300002, 300003, 700001, 700002, 700003 ≃ 20ms

10ms 10ms

Reordering requests can improve performance

SSTF (SHORTEST SEEK TIME FIRST)



Strategy always choose request that requires least seek time (approximate total time with seek time)

Greedy algorithm (just looks for best NEXT decision)

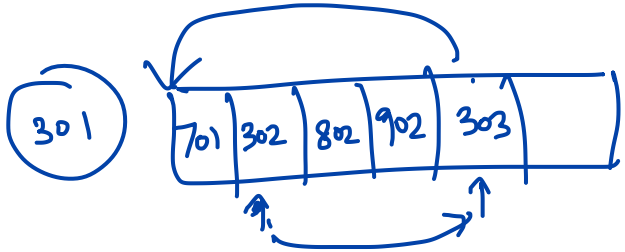
How to implement in OS? → How much seek time will it take? for a sector number

Sort sector numbers

Sort by sector number

Disadvantages?

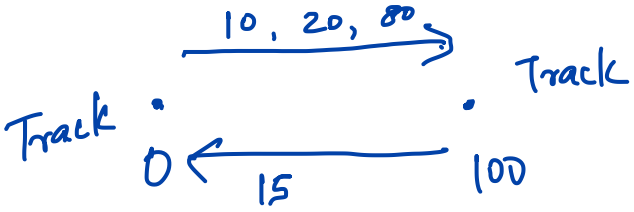
Starvation



SCAN

10, 20, 80, 15

new req 15 arrives between 20 & 80



SCAN or Elevator Algorithm:

- Sweep back and forth, from one end of disk other, serving requests as pass that cylinder
- Sorts by cylinder number; ignores rotation delays

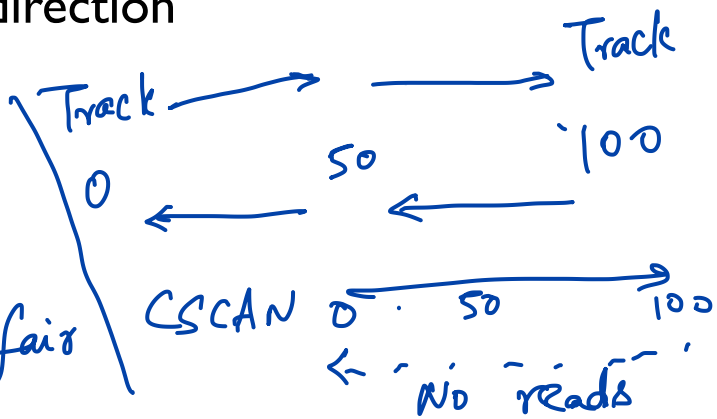
Trade-off

C-SCAN (circular scan): Only sweep in one direction

Pros/Cons?

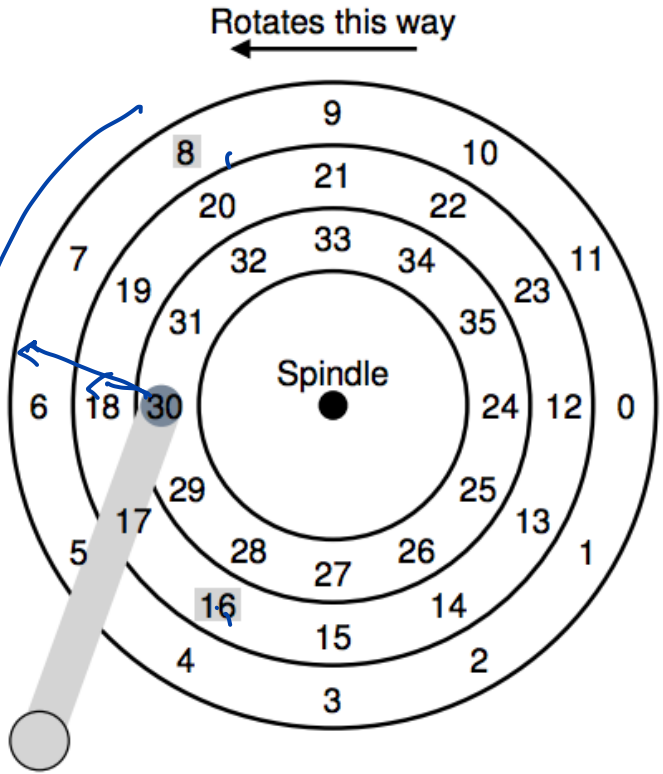
SCAN

scan the internal tracks get to go twice → Not fair



SPTF (SHORTEST POSITIONING TIME FIRST)

In the example 8 may have lower access time!

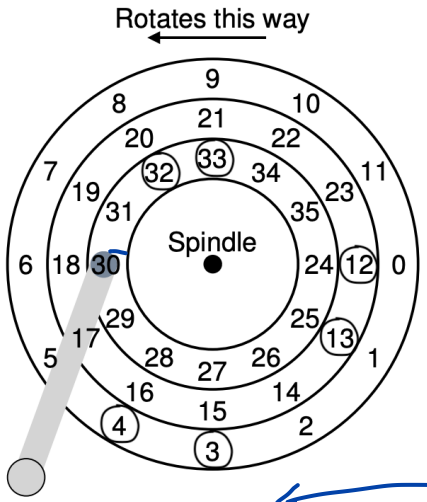


Greedy algorithm takes into account both seek & rotation costs

SATF (SHORTEST ACCESS TIME FIRST)

QUIZ 22

https://tinyurl.com/cs537-sp20-quiz22



Disk accesses: 32, 12, 33, 3, 13, 4

Rotation Time = 2ms (non-adjacent reads)

Seek Time (for adjacent track) = 2ms.

4 ms for jumping two tracks

What is the time taken to using (FCFS) scheduling?

32 : 2 ms

12 : 2 ms for seek + 2 ms for rot = 4

33 : 4 ms

3 : 4 ms + 2 = 6 → going to two tracks

13 : 2 ms + 2 ms = 4

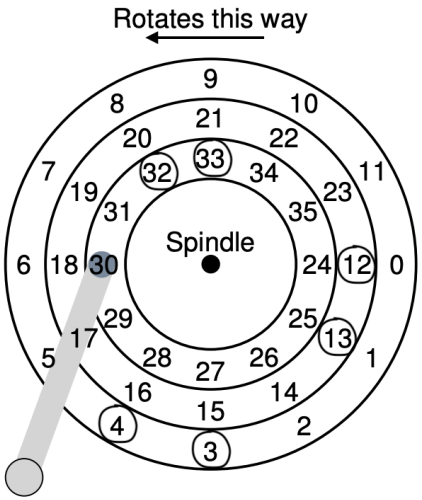
4 : 2 ms + 2 ms = 4

= 24 ms

Order in which requests will be serviced for Shortest Seek Time First (SSTF)?

QUIZ 22

https://tinyurl.com/cs537-sp20-quiz22

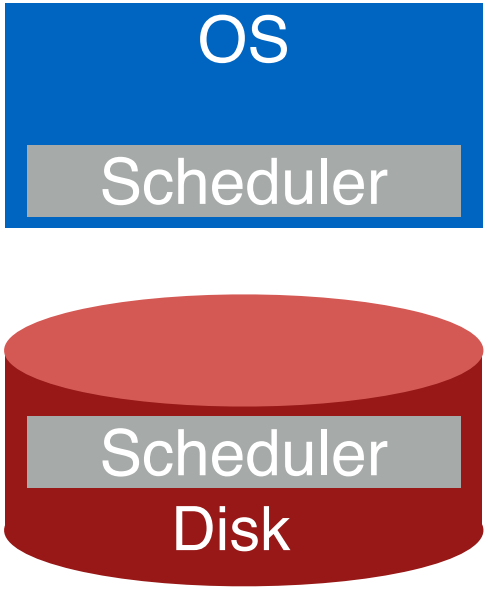


Disk accesses: 32, 12, 33, 3, 13, 4
Rotation Time = 2ms (non-adjacent reads)
Seek Time (for adjacent track) = 2ms.

Order in which requests will be serviced for Shortest Seek Time First (SSTF)?

Time Taken

SCHEDULERS



Where should the scheduler go?

WHAT HAPPENS?

Assume 2 processes each calling read() with C-SCAN

```
void reader(int fd) {
    char buf[1024];
    int rv;
    while((rv = read(fd, buf)) != 0) {
        assert(rv);
        // takes short time, e.g., 1ms
        process(buf, rv);
    }
}
```

WORK CONSERVATION

Work conserving schedulers always try to do work if there's work to be done

Sometimes, it's better to wait instead if system anticipates another request will arrive

Possible improvements from I/O Merging

SUMMARY

Disks: Specific geometry with platters, spindle, tracks, sector

I/O Time: rotation_time + seek_time + transfer_time
Sequential throughput vs. random throughput

Advanced Techniques: Skewed layout, caching

Scheduling approaches: SSTF, SCAN, C-SCAN
Benefits of violating work conservation

NEXT STEPS

Next class: How to achieve resilience against disk errors

Project 4a in Discussion today