Goals for Today

- Learning Objective:
  - Review midterm results
  - Begin our exploration of file systems!
- Announcements, etc:
  - C4: All remaining submissions open on Compass now
  - MP3 is out! Due **April 18th**.

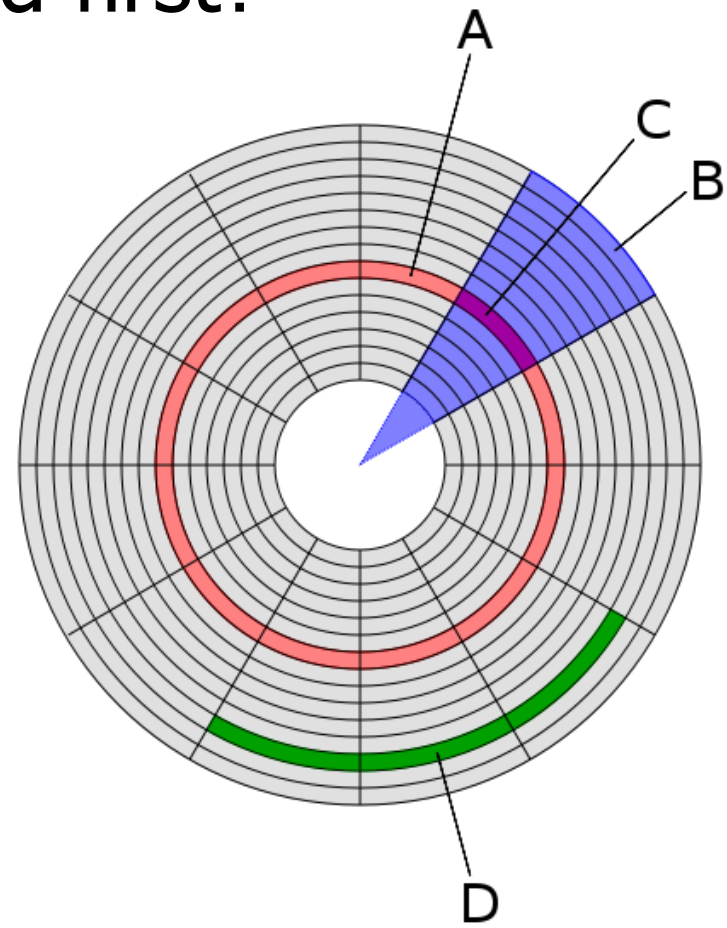
Reminder: Please put away devices at the start of class

CS 423
Operating System Design: Linux Disk Scheduling

Professor Adam Bates
Spring 2018

Disk Scheduling

- Which disk request is serviced first?
  - FCFS
  - Shortest seek time first
  - Elevator (SCAN)
  - C-SCAN (Circular SCAN)

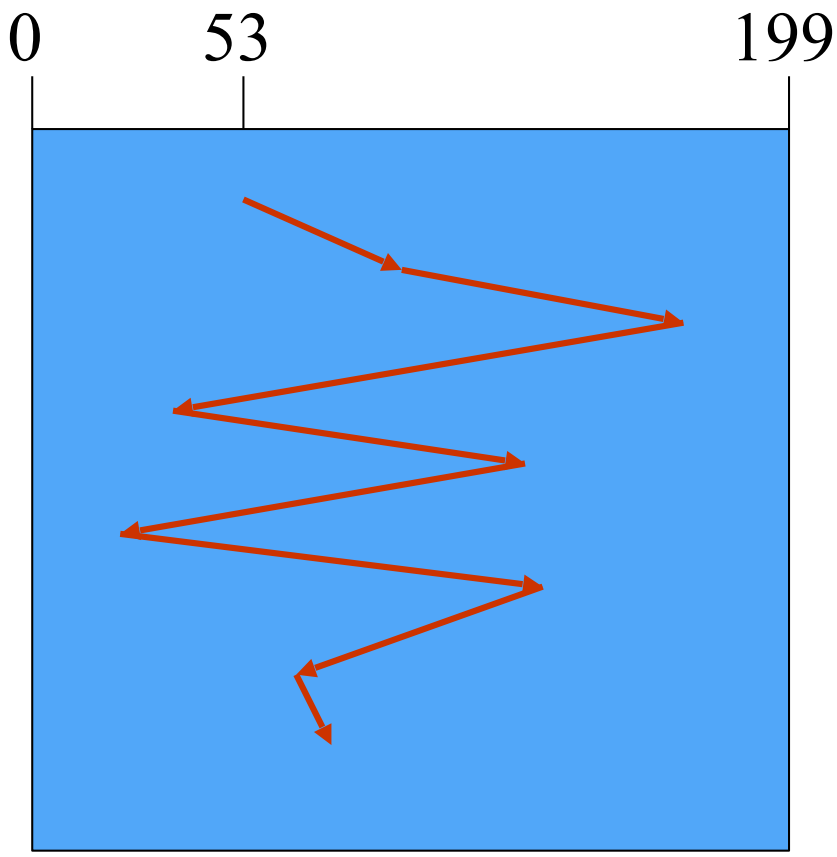


A: Track.
B: Sector.
C: Sector of Track.
D: File

Disk Scheduling Decision — Given a series of access requests, on which track should the disk arm be placed next to maximize fairness, throughput, etc?

FIFO (FCFS) Order

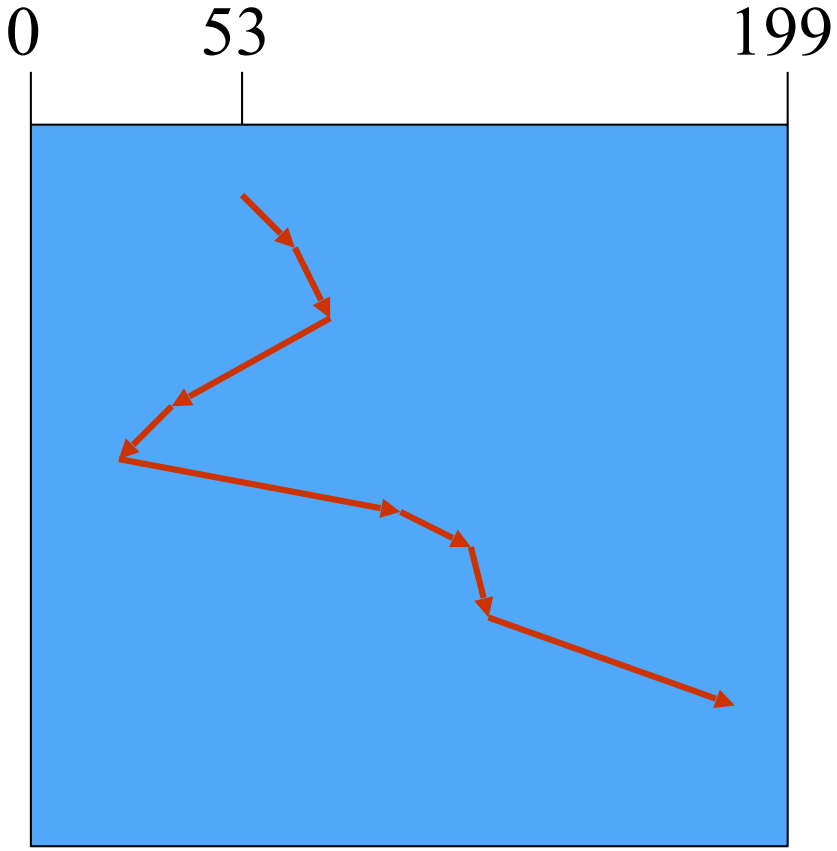
- Method
  - First come first serve
- Pros?
  - Fairness among requests
  - In the order applications expect
- Cons?
  - Arrival may be on random spots on the disk (long seeks)
  - Wild swing can happen
- Analogy:
  - FCFS elevator scheduling?



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

SSTF (Shortest Seek Time First)

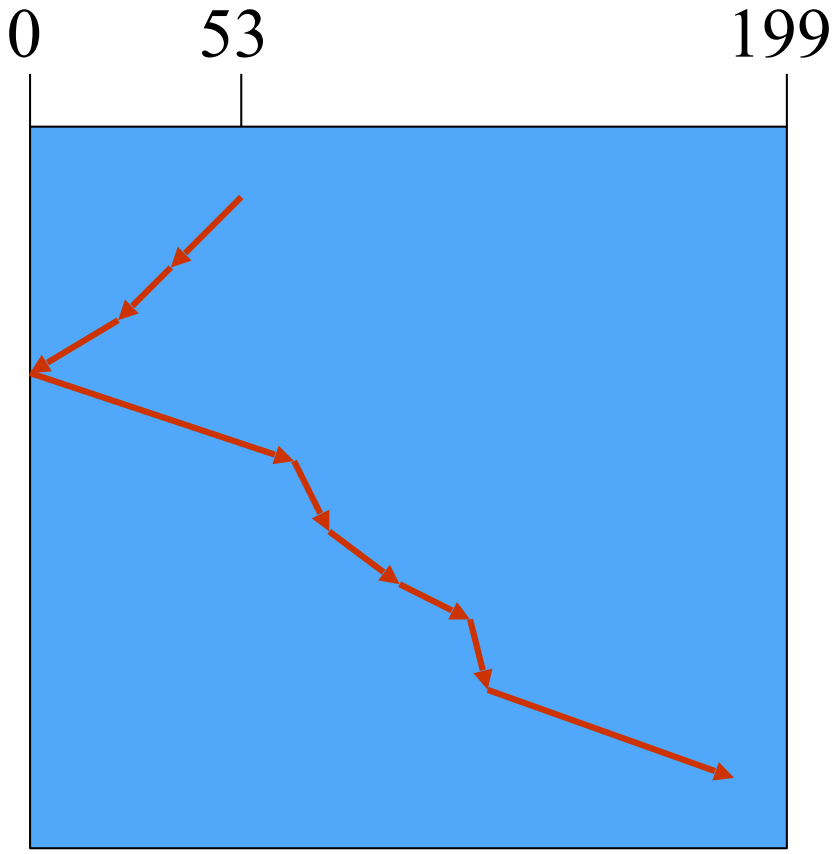
- Method
  - Pick the one closest on disk
- Pros?
  - Try to minimize seek time
- Cons?
  - Starvation
- Question
  - Is SSTF optimal?
  - Are we worried about sorting overhead?
  - Can we avoid starvation?



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

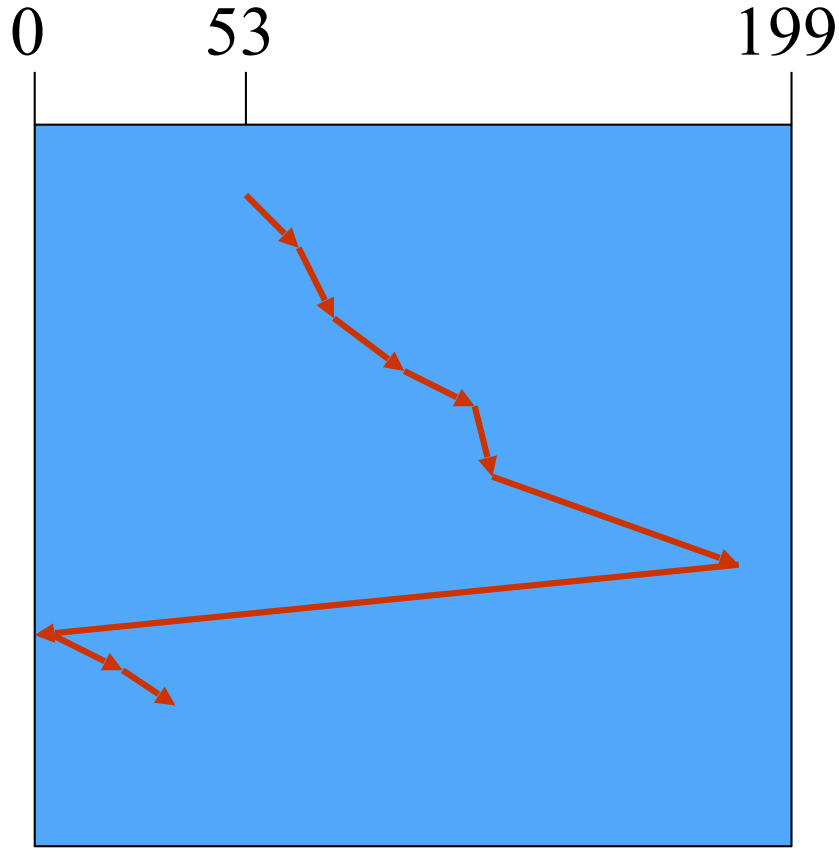
Elevator (SCAN)

- Method
  - Take the closest request in the direction of travel
- Pros
  - Bounded time for each request
- Cons?
  - Request at the other end will take a while
  - Which sectors have shorter wait times?
    - How to fix?



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

C-SCAN (Circular SCAN)



- Method
  - Like SCAN
  - But, wrap around
- Pros
  - Uniform service time
- Cons
  - Do nothing on the return

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

Scheduling Algorithms

Algorithm Name	Description
FCFS	First-come first-served
SSTF	Shortest seek time first; process the request that reduces next seek time
SCAN (aka Elevator)	Move head from end to end (has a current direction)
C-SCAN	Only service requests in one direction (circular SCAN)
LOOK	Similar to SCAN, but do not go all the way to the end of the disk.
C-LOOK	Circular LOOK. Similar to C-SCAN, but do not go all the way to the end of the disk.

Disk Scheduling Performance

- What factors impact disk performance?
  - Seek Time: Time taken to move disk arm to a specified track
  - Rotational Latency: Time taken to rotate desired sector into position
  - Transfer time: Time to read/write data. Depends on rotation speed of disk and transfer amount.

```
Disk Access Time = Seek Time +
                   Rotational Latency +
                   Transfer Time
```

Linux I/O Schedulers

- What disk (I/O) schedulers are available in Linux?

```
$ cat /sys/block/sda/queue/scheduler
noop [deadline] cfq
```

^ scheduler enabled on our VMs

- As of Linux 2.6.10, it is possible to change the IO scheduler for a given block device on the fly!
- How to enable a specific scheduler?

```
$ echo SCHEDNAME > /sys/block/DEV/queue/scheduler
```

  - SCHEDNAME = Desired I/O scheduler
  - DEV = device name (e.g., hda)

Linux NOOP Scheduler

- `NOOP`
  - Insert all incoming I/O requests into a simple FIFO
  - Merges duplicate requests (results can be cached)
- When would this be useful?

Linux NOOP Scheduler

- Insert all incoming I/O requests into a simple FIFO
- Merges duplicate requests (results can be cached)
- When would this be useful?
  - Solid State Drives! Avoids scheduling overhead
  - Scheduling is handled at a lower layer of the I/O stack (e.g., RAID Controller, Network-Attached)
  - Host doesn't actually know details of sector positions (e.g., RAID controller)

Linux Deadline Scheduler

- Imposes a deadline on all I/O operations to prevent starvation of requests
- Maintains 4 queues:
  - 2 Sorted Queues (R, W), order by Sector
  - 2 Deadline Queues (R, W), order by Exp Time
- Scheduling Decision:
  - Check if 1st request in deadline queue has expired.
  - Otherwise, serve request(s) from Sorted Queue.
  - Prioritizes reads (DL=500ms) over writes (DL=5s) .Why?

Linux CFQ Scheduler

- CFQ = Completely Fair Queueing!
- Maintain per-process queues.
- Allocate time slices for each queue to access the disk
- I/O Priority dictates time slice, # requests per queue
- Asynchronous requests handled separately — batched together in priority queues
- CFQ is normally the default scheduler

Linux Anticipatory Scheduler

- Deceptive Idleness: A process appears to be finished reading from disk, but is actually processing data. Another (nearby) request is coming soon!
- Bad for synchronous read workloads because seek time is increased.
- Anticipatory Scheduling: Idle for a few milliseconds after a read operation in *anticipation* of another close-by read request.
- Deprecated — LFQ can approximate.