Elevator (Disk) Scheduling

COMPSCI 210 Recitation

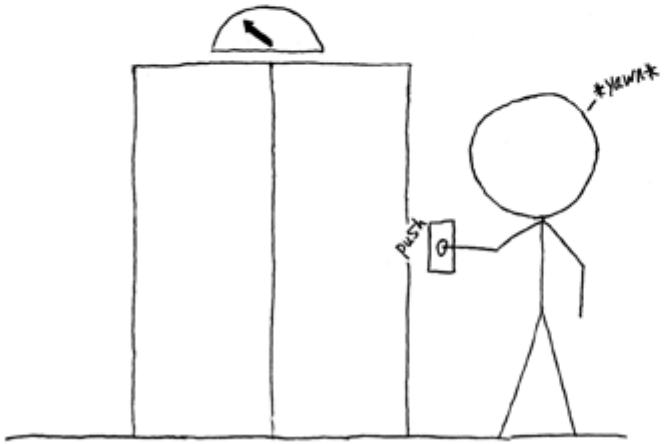
26th Oct 2012

Vamsi Thummala

What metrics do you consider for elevator scheduling?

MOST PEOPLE PRESS THE ELEVATOR CALL BUTTON ONCE AND WAIT PATIENTLY.

THAT IS WRONG.



I PRESS THE BUTTON MULTIPLE TIMES IN RAPID SUCCESSION.

THE ELEVATOR SENSES MY FRUSTRATION AND SERVES ME WITH GREAT HASTE.

http://www.elevatorworld.com/blogs/?p=1214

Metrics for elevator scheduling

- Service time
  - Time between pushing the button and exit the elevator
  - Approximate
    - Wait time
- Fairness
  - Variation in the service time(s)
- Efficiency
  - Roughly defined as the amount of total work done (Energy)
  - Work done: Number of floors the elevators pass in total
- Objective: Minimize service time, Maximize fairness, Minimize work done

Elevator data structure(s)

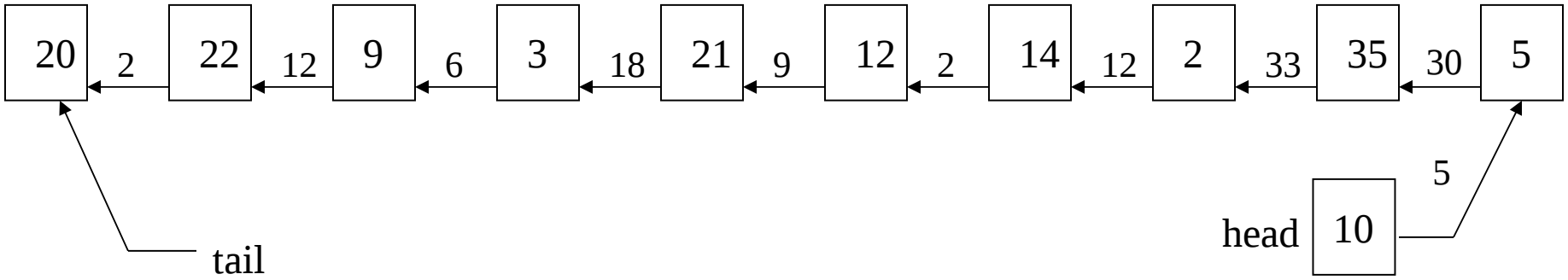
- ElevatorController
  - Pool/Queue of events
    - CallUp/CallDown
- Elevator
  - Pool/Queue of requests
    - Direction
    - Destination floor

First Come First Served (FCFS)

- Service in the order in which the requests are made
  - Riders enter and press the destination floor
- Simple to implement
- No starvation
  - Every request is serviced
- Is FCFS a good policy?

FCFS

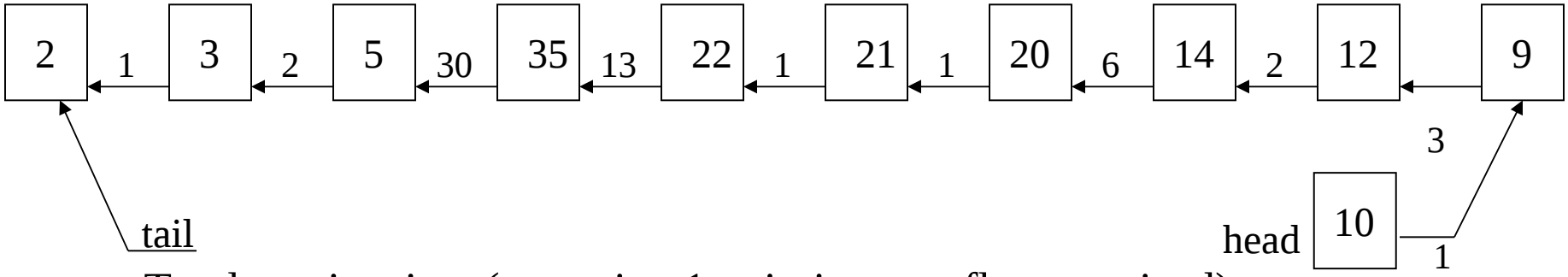
- The elevator is currently servicing the 10th floor
- Order of requests from riders at the 10th floor:
  5 (down), 35 (up), 2 (down), 14 (up), 12 (up), 21 (up),
  3 (down), 9 (down), 22 (up), 20 (up)
- To simplify, let us assume everyone gets in



- Total service time (assuming 1 unit time per floor serviced):
  5 + 30 + 33 + 12 + 2 + 9 + 18 + 6 + 12 + 2 = 129,
  Avg service time: 12.9
- Can we do better?
  - Service the closest floor

Shortest Seek Time First (SSTF)

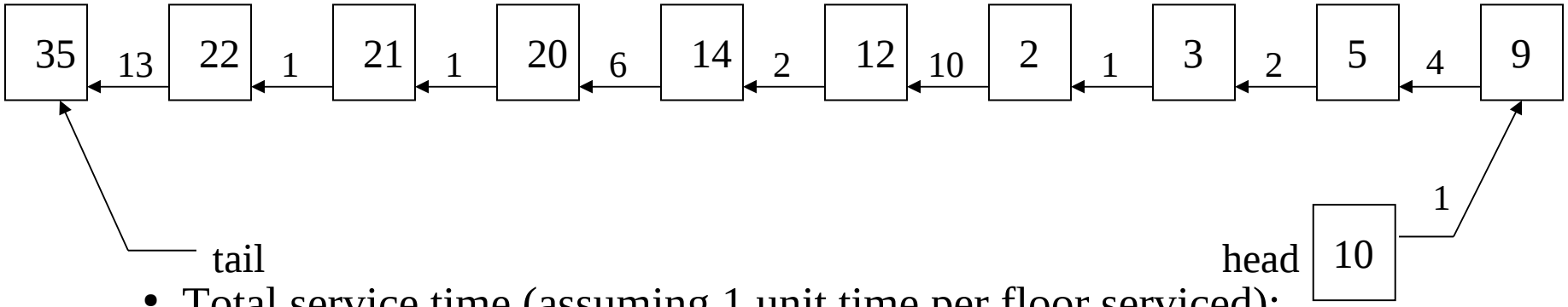
- Go to the closest floor in the work queue
- Reduces total seek time compared to FCFS
- Order of requests from riders at the 10th floor:
  5 (down), 35 (up), 2 (down), 14 (up), 12 (up), 21 (up),
  3 (down), 9 (down), 22 (up), 20 (up)



- Total service time (assuming 1 unit time per floor serviced):
  1 + 3 + 2 + 6 + 1 + 1+ 13 + 30 + 2 + 1 = 60, Avg: 6
- Disadvantages:
  - Starvation possible
  - Switching directions may slow down the actual service time
- Can we do better? Reorder the requests w.r.t direction

SCAN

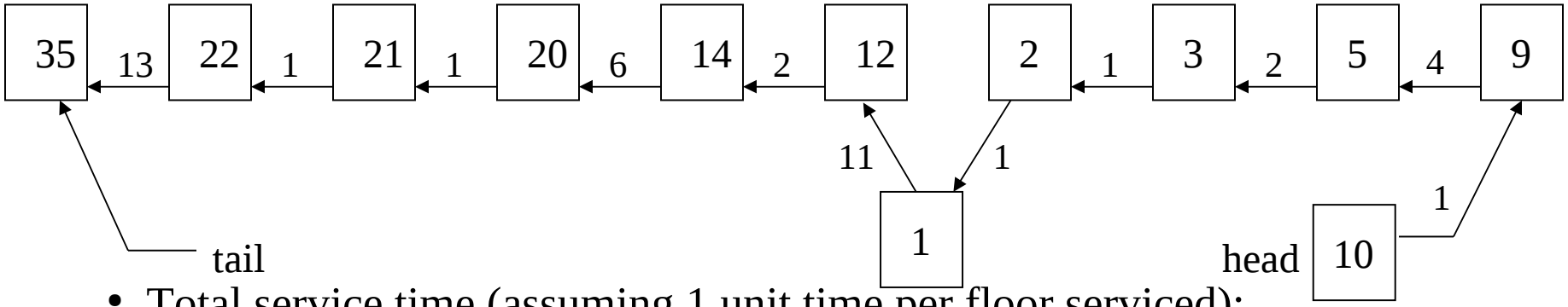
- Start servicing in a given direction to the end
  - Change direction and start servicing again
- Order of requests from riders at the 10th floor:
  5 (down), 35 (up), 2 (down), 14 (up), 12 (up), 21 (up),
  3 (down), 9 (down), 22 (up), 20 (up)



- Total service time (assuming 1 unit time per floor serviced):
  1 + 4 + 2 + 1 + 10 + 2 + 6 + 1 + 1 + 13 = 41, Avg: 4.1
- Advantages
  - Reduces variance in seek time
- Can we do better?

Circular SCAN (C-SCAN)

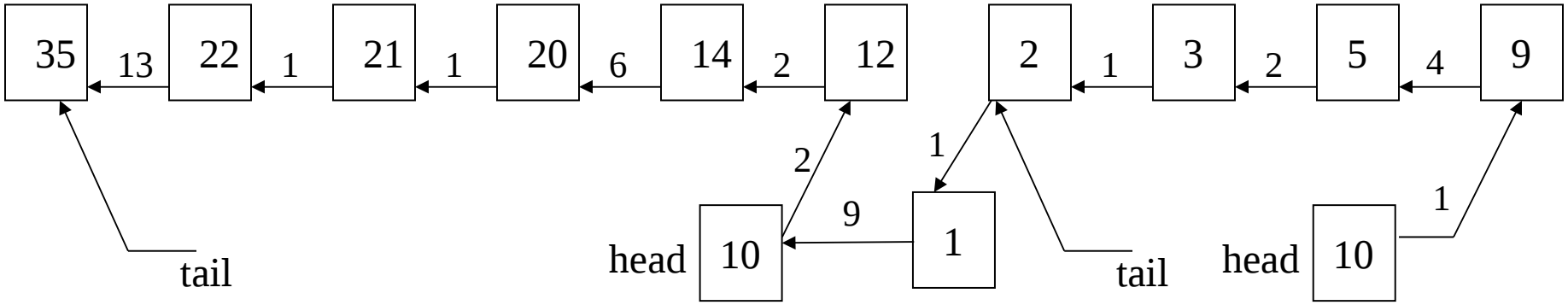
- Start servicing in a given direction to the end
  - Go to the first floor without servicing any requests; Restart servicing
- Order of requests from riders at the 10th floor:
  5 (down), 35 (up), 2 (down), 14 (up), 12 (up), 21 (up),
  3 (down), 9 (down), 22 (up), 20 (up)



- Total service time (assuming 1 unit time per floor serviced):
  1 + 4 + 2 + 1 + 1 + 11 + 2 + 6 + 1 + 1 + 13 = 43, Avg: 4.3
- Advantages
  - More fair compared to SCAN
- Is this what you expect in a real-world elevator?

Elevator Scheduling

- At least one difference from C-SCAN
  - Direction of pick up
- Order of requests from riders at the 10th floor:
  5 (down), 35 (up), 2 (down), 14 (up), 12 (up), 21 (up),
  3 (down), 9 (down), 22 (up), 20 (up)

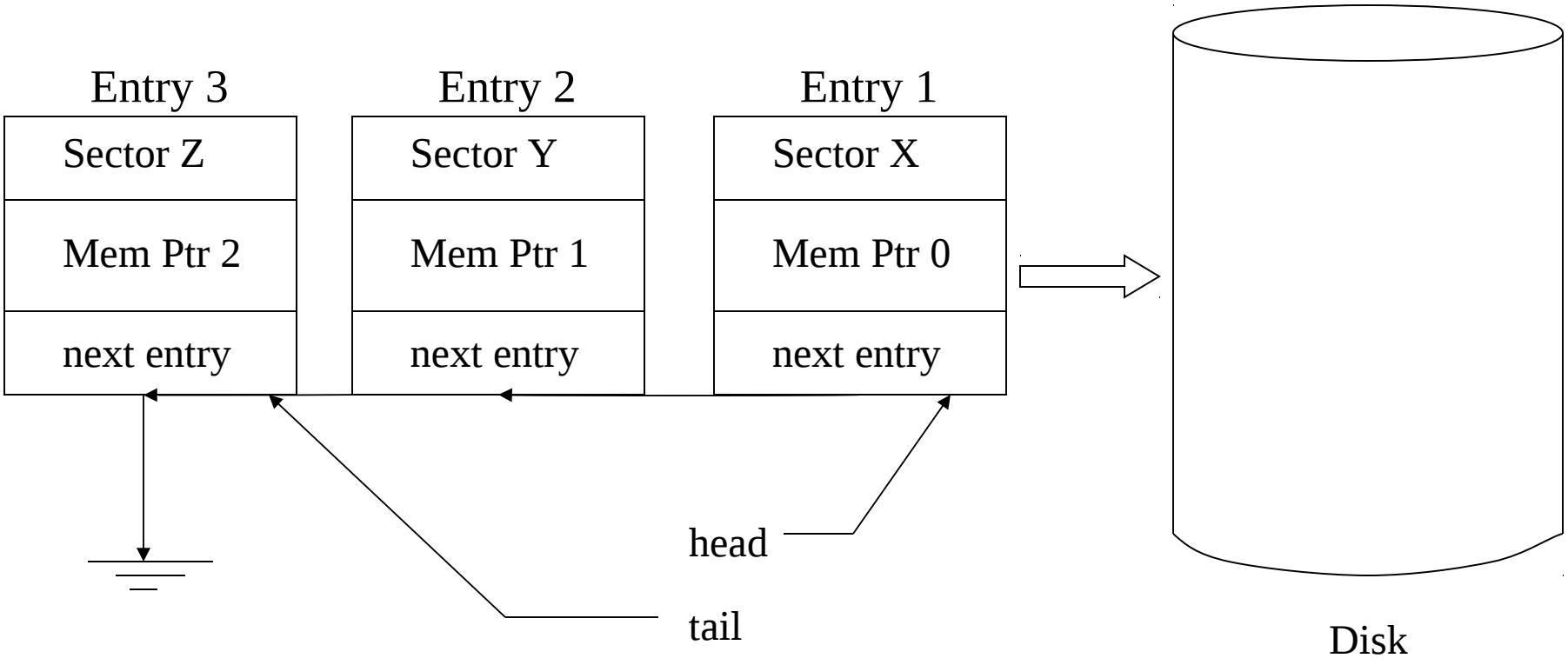


- Total service time (assuming 1 unit time per floor serviced):
  1 + 4 + 2 + 1 + 1 + 9 + 2 + 2 + 6 + 1 + 1 + 13 = 43, Av: 4.3
- Can you do better?
  - We look forward to your lab submissions

Disk Scheduling

- Similar to elevator scheduling
- Each disk has a queue of jobs waiting to access disk
  - read jobs
  - write jobs
- Each entry in queue contains the following
  - pointer to memory location to read/write from/to
  - sector number to access
  - pointer to next job in the queue
- OS usually maintains this queue

Disk Queues


Entry 3
Entry 2
Entry 1
Sector Z
Sector Y
Sector X
Mem Ptr 2
Mem Ptr 1
Mem Ptr 0
next entry
next entry
next entry
head
tail
Disk

CPU Scheduling

- Monday's lecture (10/29)
- More variations
  - Priority
  - Round robin
  - Preemption
  - Multilevel queue

Back to synchronization

- Spring 2001 Midterm

  Operators are standing by. The customer service line at MegaMurk staffs a phone bank to answer questions from customers about its products. Customers calling in are put on hold, where they wait for the next available customer service advocate to answer their call "in the order it was received". The economy is in a downturn, so business is sometimes slow: if there are no customers waiting, available advocates just wait around for the phone to ring. You are to write procedures to correctly synchronize the customers and advocates. You need not be concerned with starvation, fairness, or deadlock.

  (a) Synchronize customers and advocates using mutexes and condition variables.

  (b) Synchronize customers and advocates using only semaphores.