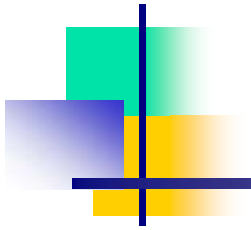


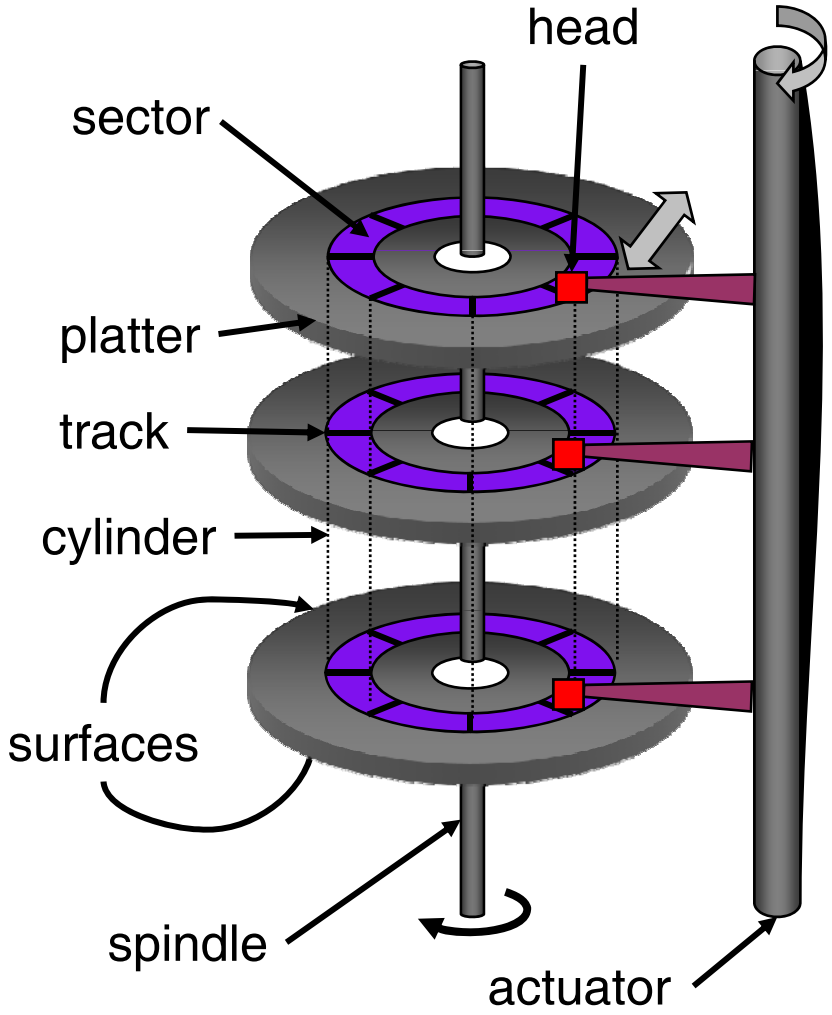
Operating Systems

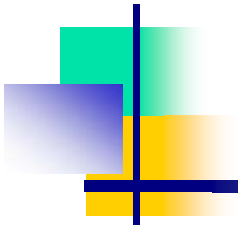
Lecture3.3 - Disk Management

By: Hossein Momeni
hmomeni@gmail.com

Disk drive structure

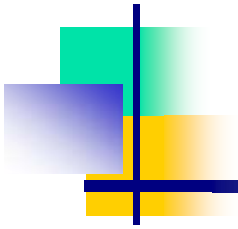
- Data stored on surfaces
  - Up to two surfaces per platter
  - One or more platters per disk
- Data in concentric tracks
  - Tracks broken into sectors
    - 256B-1KB per sector
  - Cylinder: corresponding tracks on all surfaces
- Data read and written by heads
  - Actuator moves heads
  - Heads move in unison





What's in a disk request?

- Time required to read or write a disk block determined by 3 factors
  - Seek time (the time to move disk arm to desired cylinder)
  - Rotational time (the time for desired sector to rotate under the disk head)
  - Transfer time
    - Transfer time = time to transfer data

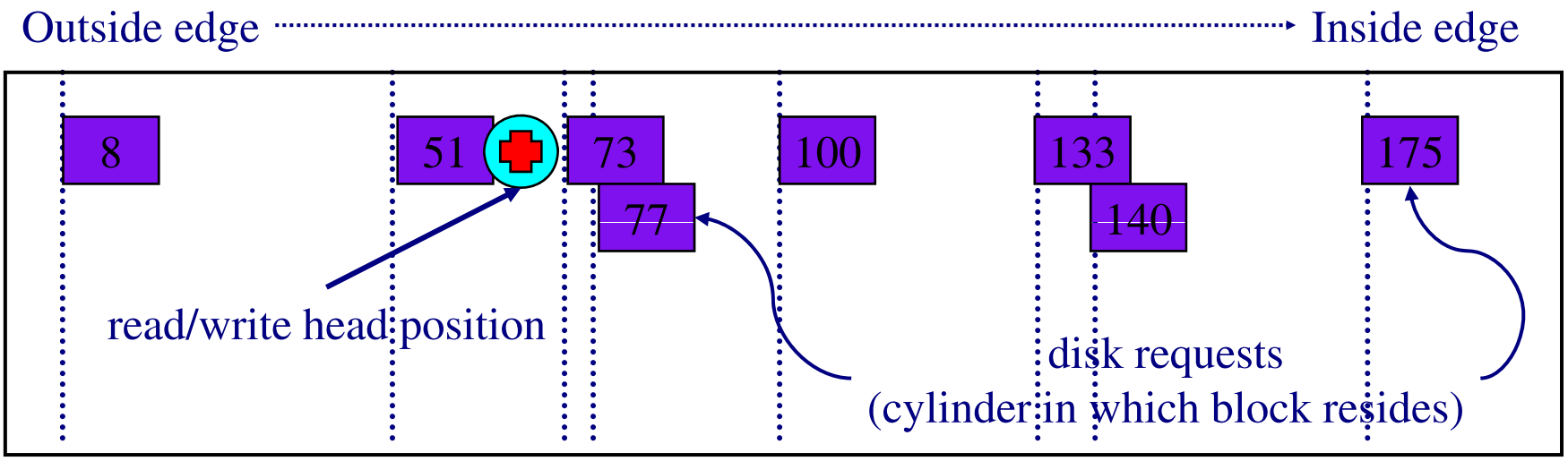


Disk request scheduling

- Goal: use disk hardware efficiently
  - Bandwidth as high as possible
  - Disk transferring as often as possible (and not seeking)
- We want to
  - Minimize disk seek time (moving from track to track)
  - Minimize rotational latency (waiting for disk to rotate the desired sector under the read/write head)
- Calculate disk bandwidth by
  - Total bytes transferred / time to service request
  - Seek time & rotational latency are overhead (no data is transferred), and reduce disk bandwidth
- Minimize seek time & rotational latency by
  - Using algorithms to find a good sequence for servicing requests
  - Placing blocks of a given file "near" each other

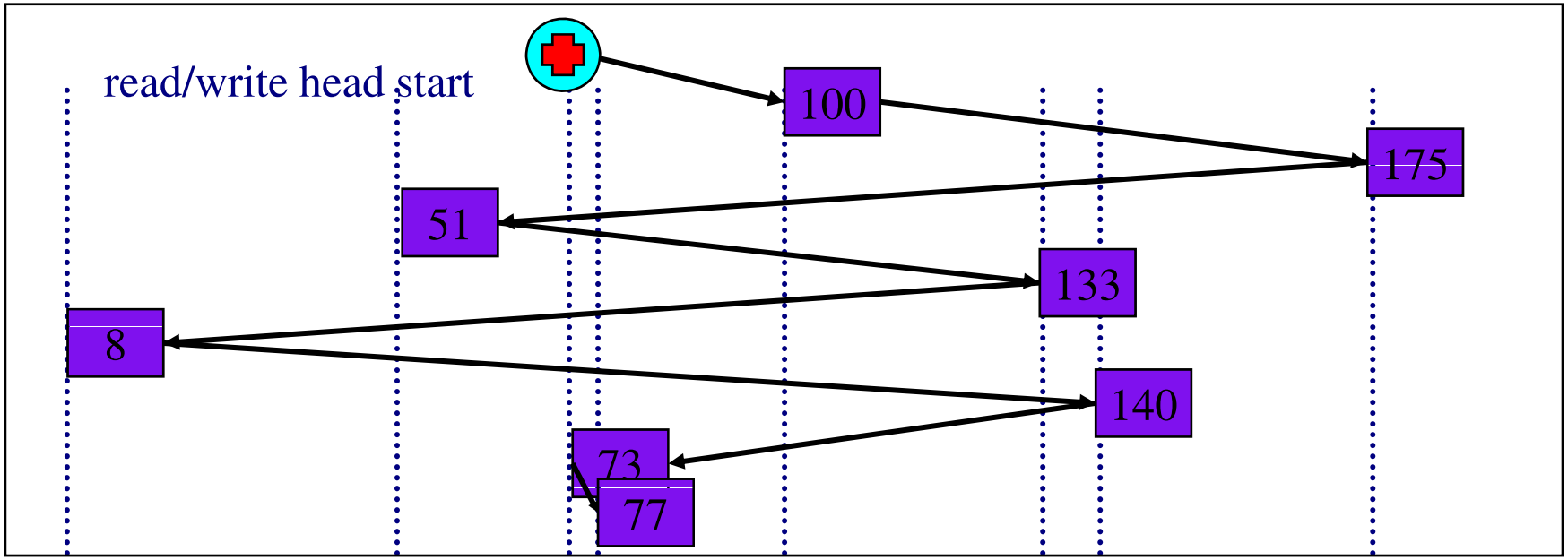
Disk Arm scheduling algorithms

- Schedule disk requests to minimize disk seek time
  - Seek time increases as distance increases (though not linearly)
  - Minimize seek distance -> minimize seek time
- Disk seek algorithm examples assume a request queue & head position (disk has 200 cylinders)
  - Queue = 100, 175, 51, 133, 8, 140, 73, 77
  - Head position = 63



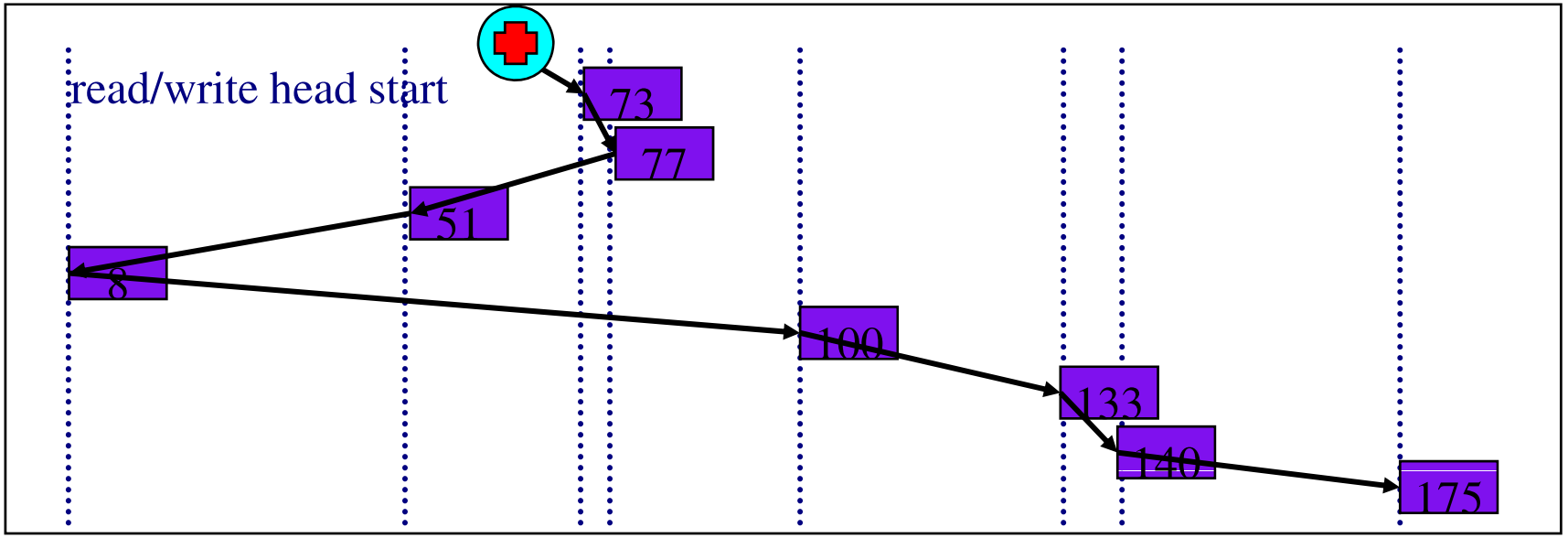
First-Come-First Served (FCFS)

- Requests serviced in the order in which they arrived
  - Easy to implement!
  - May involve lots of unnecessary seek distance
- Seek order = 100, 175, 51, 133, 8, 140, 73, 77
- Seek distance = (100-63) + (175-100) + (175-51) + (133-51) + (133-8) + (140-8) + (140-73) + (77-73) = 646 cylinders



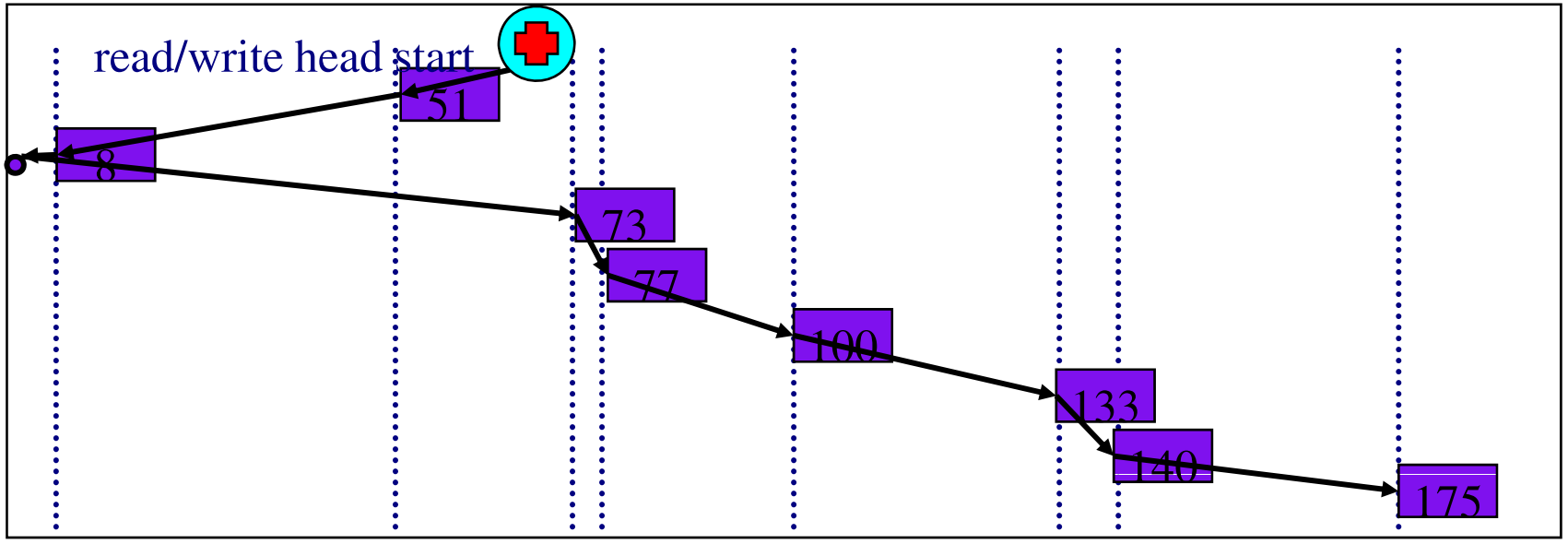
Shortest Seek Time First (SSTF)

- Service the request with the shortest seek time from the current head position
  - Form of SJF scheduling
  - May starve some requests
- Seek order = 73, 77, 51, 8, 100, 133, 140, 175
- Seek distance = 10 + 4 + 26 + 43 + 92 + 33 + 7 + 35 = 250 cylinders



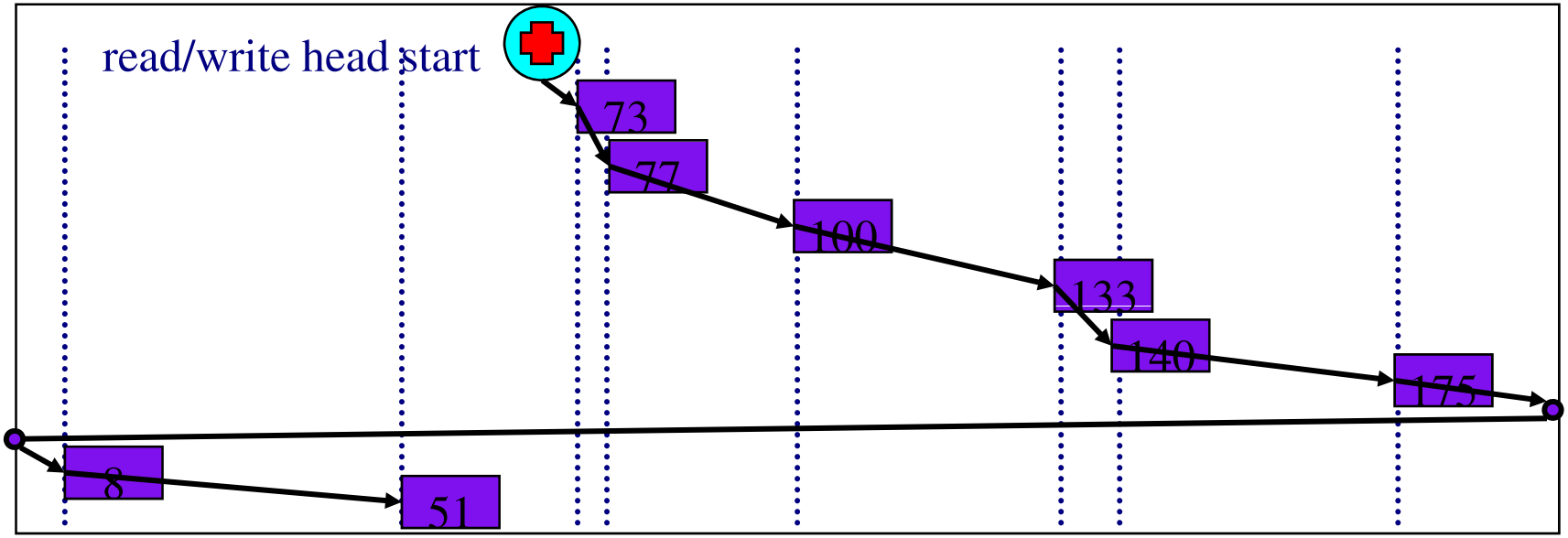
SCAN (elevator algorithm)

- Disk arm starts at one end of the disk and moves towards the other end, servicing requests as it goes
  - Reverses direction when it gets to end of the disk
  - Also known as elevator algorithm
- Seek order = 51, 8, 0 , 73, 77, 100, 133, 140, 175, 199
- Seek distance = 12 + 43 + 8 + 73 + 4 + 23 + 33 + 7 + 35+24 = 262 cyls



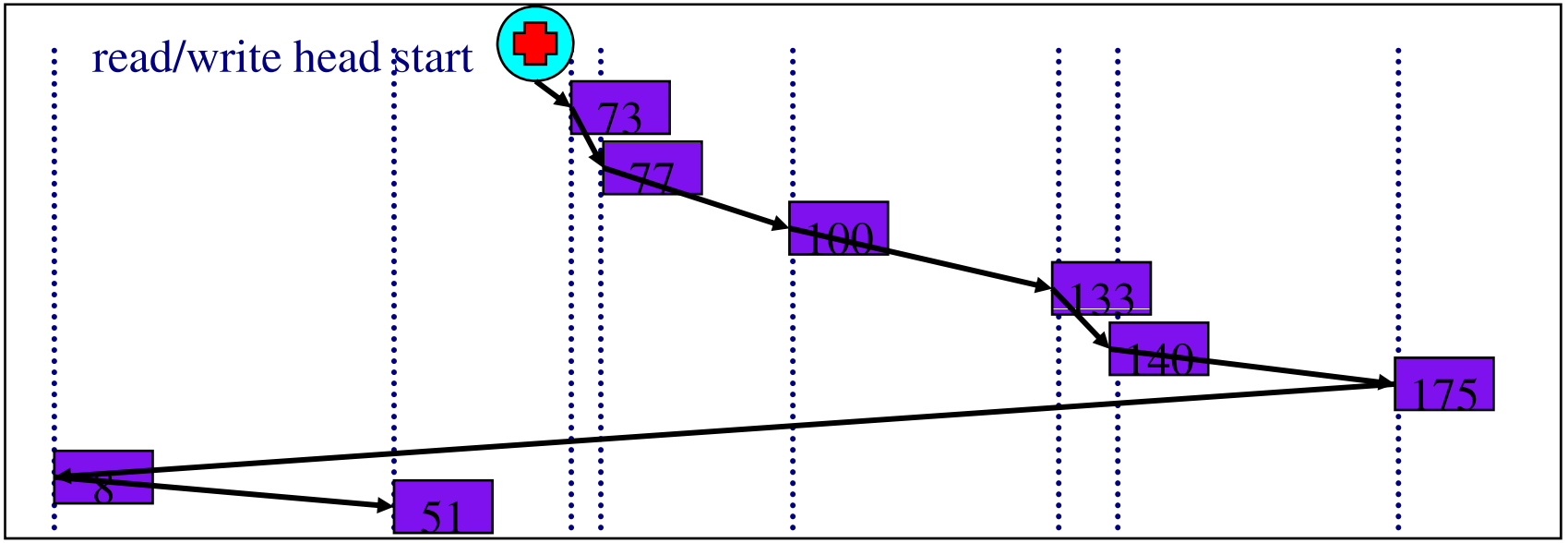
C-SCAN (Circular scan)

- Identical to SCAN, except head returns to cylinder 0 when it reaches the end of the disk
  - Treats cylinder list as a circular list that wraps around the disk
  - Waiting time is more uniform for cylinders near the edge of the disk
- Seek order = 73, 77, 100, 133, 140, 175, 199, 0, 8, 51
- Distance = 10 + 4 + 23 + 33 + 7 + 35 + 24 + 199 + 8 + 43 = 386 cyls



C-LOOK

- Identical to C-SCAN, except head only travels as far as the last request in each direction
  - Saves seek time from last sector to end of disk
- Seek order = 73, 77, 100, 133, 140, 175, 8, 51
- Distance = 10 + 4 + 23 + 33 + 7 + 35 + 167 + 43 = 322 cylinders



How to pick a disk scheduling algorithm

- SSTF is easy to implement and works OK if there aren't too many disk requests in the queue
- SCAN-type algorithms perform better for systems under heavy load
  - More fair than SSTF
  - Use LOOK rather than SCAN algorithms to save time
- Long seeks aren't too expensive, so choose C-LOOK over LOOK to make response time more even
- Disk request scheduling interacts with algorithms for allocating blocks to files
  - Make scheduling algorithm modular: allow it to be changed without changing the file system

⇒ Use SSTF for lightly loaded systems

⇒ Use C-LOOK for heavily loaded systems