I/O Menegemnt

How meny I/o Devices very?

- * Data rates
- * Synchronizction
- * Data sizes (blocks, streams)
- * Error Conditions
- * Error Checking (parity bits)

[1] Program I/o [busy wait]

[2] Using Interupts: Waiting process are blocked until I/o completes

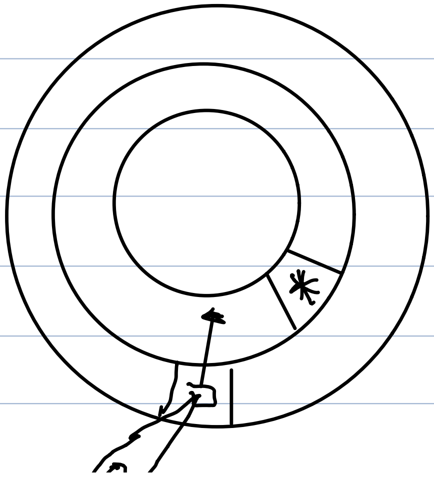
[3] Direct memory Accesss:

Device has direct access to memory

Disk Scheduling

Disk roteting at a constent speed.

To reed & write, the heed most be on the right track and at the beginning of the sector



Average Access Time

Seek time : to move head to the required track

Rotational Delay : For the addressed area to be in position accessible by he

Transfer Time : Time to read / write and transfer data.

Calculations

- Average Access Time : Seek Time (T_s)

  +

  Rotational Time (T_r)

  +

  Transfer Time (T_f)

  +

  Queuing Delay

# of bytes To read or write

$$T_f = \frac{b}{rN} \Leftarrow$$

Rotational speed

# of bytes on a track

T_s = 4 msec

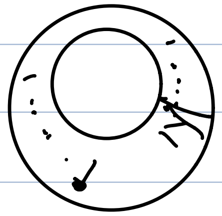
r = 7500 r/min   125 r/sec => 8 msec. each rotation

512 byte sectors track has 500 sectors

$\boxed{1}$ Read 2500 Sectors Sequentially

First Track = $T_s + T_r + T_f$

$4 + \frac{8}{2} + 8 = 16$



2 due to

the rotations

- 1st sector under the head
- 2nd for reading the full sector.

$$\text{Avg. Access Time} = \underset{\substack{\uparrow \\ \text{First} \\ \text{Track}}}{16} + 4 * (\overset{(8/2 + 8)}{\overset{\downarrow}{12}})$$

= 0.064 seconds

64ms

$\boxed{2}$ Read 2500 sectors (random)

Ts + seek time + For 1 rotation

$$\text{Avg. Access Time} = 2500 * \left(4 + \frac{8}{2} + \underset{\substack{\uparrow \\ \text{To read} \\ \downarrow \\ 0.016}}{\frac{8}{500}}\right)$$

For 1 rotation Takes 8ms

or

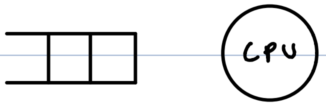
$$\frac{b = 512}{\frac{125}{1000} * 512 * 500}$$

= 2500 * 8.016

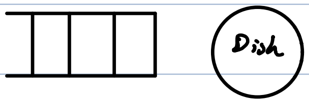
= 20 seconds

Goal :

Reduce the seek time



OS Maintains a queue holding the tracks requested



1 FIFO

process request in order recieved

If we have many processes it approaches random

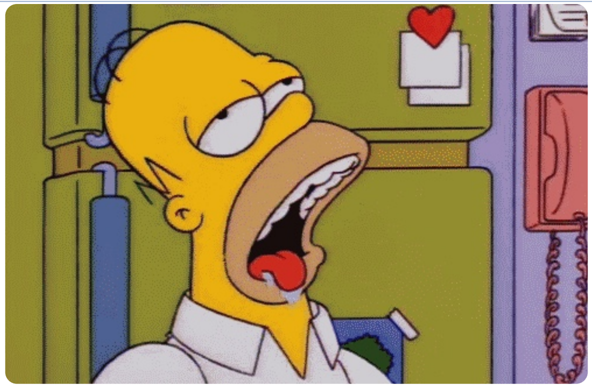
(Dependent on how data is stored)

2 Shortest Seek Time First (SSTF)

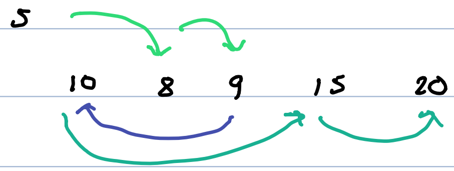
Select the nearest track to where the head is

* Lead to starvation

* Is it optimal?



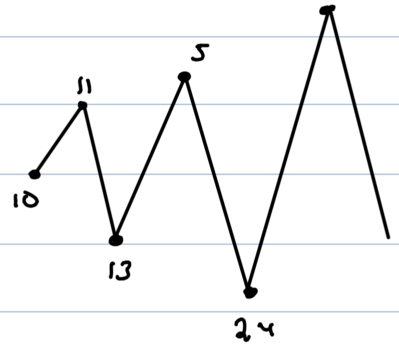
call



$5 \xrightarrow{3} 8 \xrightarrow{1} 9 \xrightarrow{1} 10 \xrightarrow{5} 15 \xrightarrow{5} 20$

- Not optimal due to another scheduler that performs

| better



Initial = 10 | 11 5 13 24

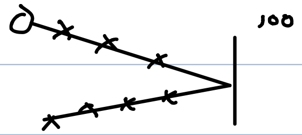
SSTF $10 \xrightarrow{1} 11 \xrightarrow{2} 13 \xrightarrow{8} 5 \xrightarrow{19} 24 \} \ 30$

$10 \xrightarrow{5} 5 \xrightarrow{6} 11 \xrightarrow{2} 13 \xrightarrow{11} 24 \} \ 24$

still a faster / opti

3 SCAN

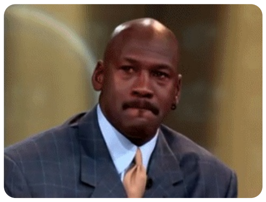
Arm moves in one direction serving request until it reaches the last track, then switches direction servicing request.

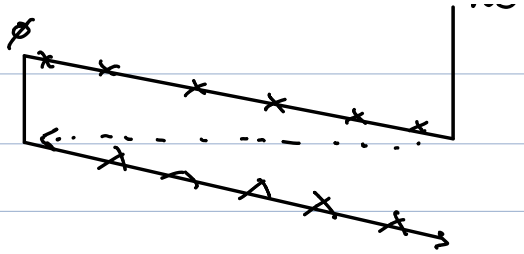


4 C SCAN

Like scan, but only service request in one direction.

107

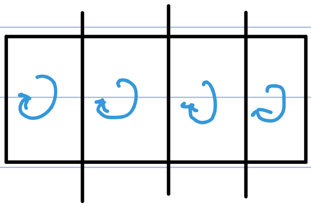




Look

With look, you move the arm to the last request.

5 N- Step Scan



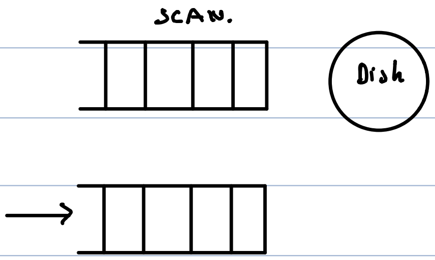
Break the queue into blocks of size N.

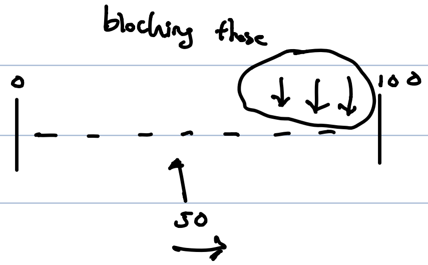
Peform scan on each block

✓ Locality

6 F SCAN.

Two queues




blocking those
0
100
50