Disk Scheduling

ECE595
Nov 27

Y. Charlie Hu

Course evaluation

- Online Course Evaluation Survey
  - Open till Sunday, Nov 9

Quiz on mmap()

The generic version of the system call used to set up a memory-mapped file looks like this:

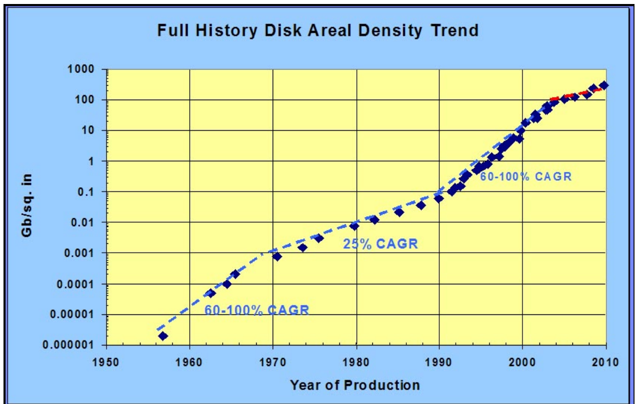
mmap(void *start_address, size_t length, int protection, int flags, int fd, off_t offset)

After an invocation of mmap, with length = 16385 bytes, successfully returns, how many physical pages have been allocated to the virtual address segment being mapped?

Typical Disks

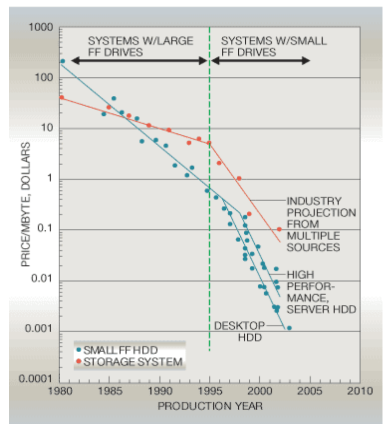
Form factor:
.5-1"× 4"× 5.7"
Storage:
18 GB - 1 TB

Form factor:
.4-.7" × 2.7" × 3.9"
Storage:
4 - 500GB



Source credit: “Technological impact of magnetic hard disk drives on storage systems”, Grochowski & Halem, IBM Systems Journal Vol 42, number 2, 2003

Figure 6 Cost of storage at the disk drive and system level

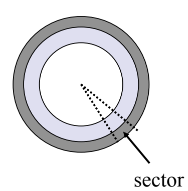


Disk Technology Trends

- Disk data is getting denser
  - More bits/square inch
  - Tracks are closer together
  - Head closer to surface
  - Density doubles every 18 months
- Disks are getting cheaper ($/MB)
  - Factor of ~2 drop per year since 1991
- Yet RPM remains largely unchanged!
  - ME is hard

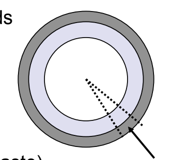
Disk Organization

- Disk surface
  - Circular disk coated with magnetic material
- Tracks
  - Concentric rings around disk surface, bits laid out serially along each track
- Sectors
  - Each track is split into arc of track (min unit of transfer)

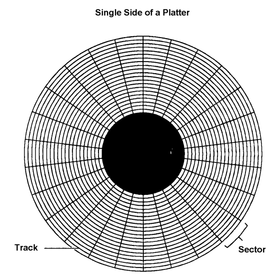


CLV vs. CAV (accessing a sector in fixed time)

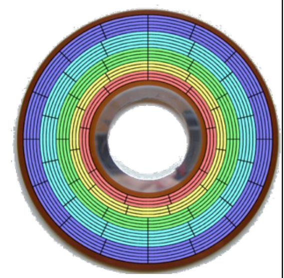
- Constant linear velocity (CLV)
  - Used in CD-ROM / DVD-ROM
  - Uniform bits density
  - The further away from center, the more sectors/track
  - Rotation speed increases moving towards center (not too bad for CD/DVD)
- Constant angular velocity (CAV)
  - Constant rotation speed
  - Bits density decreases in outer tracks (waste)
  - Used in old hard disks



Zone CAV (Zone Bit Recording)



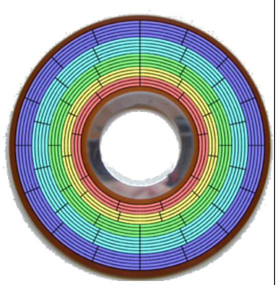
Old hard drive – CHS (Cylinder-Head-Sector)



Zoned bit recording – LBA (Linear block addressing)

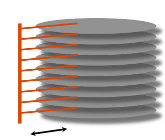
Zoned Bit Recording

- Tracks are grouped into zones based on distance from center
- Each zone has same number of sectors/track
- Best of two worlds
  - Increase the capacity
  - Constant angular velocity of the platters
  - Diff zones have similar density
- Hard disk controller implements ZBR by varying the rate it reads and writes – faster on outer cylinders (zones)



More on Disks

- CD's and floppies come individually, but magnetic disks come organized in a disk pack
- Cylinder
  - Certain track of the platter
- Disk arm
  - Seek the right cylinder



Disk Examples (Summarized Specs)

	Seagate Barracuda	IBM Ultrastar 72ZX
Capacity, Interface & Configuration		
Formatted Gbytes	1 TB	73.4
Interface	USATA 6Gb/s	Ultra160 SCSI
Spindle speed (RPM)	7200	10000
Bytes per sector	512	512-528
Performance		
Max Internal transfer rate (Mbytes/sec)		53
Max external transfer rate (Mbytes/sec)	600	160
Avg Transfer rate (Mbytes/sec)	125	22.1-37.4
Cacher (Mbytes)	32	16
Average seek, read/write (msec)	<8.5/9.5	5.3
Average rotational latency (msec)	4.17	2.99
Spindle speed (RPM)	7,200	10,000

Internal transfer rate: between platters and disk's integrated controller
External transfer rate: between disk and the rest of the PC

Disk Performance

- Seek
  - Position heads over cylinder, typically **5.3 - 8 ms**
- Rotational delay
  - Wait for a sector to rotate underneath the heads
  - Typically 8.3 - 6.0 ms (7,200 – 10,000RPM) or ½ rotation takes **4.15-3ms**
- Transfer bandwidth
  - Average transfer bandwidth (**15-37 Mbytes/sec**)

- Performance of transfer 1 Kbytes
  - Seek (5.3 ms) + half rotational delay (3ms) + transfer (0.04 ms)
  - Total time is 8.34ms → **120 Kbytes/sec**!

- What block size can get 90% of the disk transfer bandwidth?

Disk Behaviors

- Seek time and rotational latency dominates the cost of small reads
  - A lot of disk transfer bandwidth are wasted

Block Size (Kbytes)	% of Disk Transfer Bandwidth
1Kbytes	0.5%
8Kbytes	3.7%
256Kbytes	55%
1Mbytes	83%
2Mbytes	90%

Observations

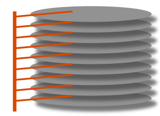
- Getting first byte from disk read is slow
  - high ***latency***
- Peak disk bandwidth good, but rarely achieved

- How to mitigate disk performance impact?
  - Move some disk data into main memory – file caching
  - Do extra calculations to speed up disk access
    - There are often multiple disk requests outstanding
    - Schedule requests to shorten seeks!

Roadmap

- Functionality (API)
  - Basic functionality
    - Disk layout
    - File operations (open, read, write, close)
  - Directories
- Performance
  - Disk allocation
  - Buffer cache
  - File System interface
  - Disk scheduling
- Reliability
  - FS level
  - Disk level: RAID

Disk Scheduling



- Assumption:
  - 1-dimental array of logical blocks is mapped to the sectors of disk sequentially
    - Sector 0 is 1st section of 1st track of outermost cylinder
    - Mappings proceeds in that track
    - Then in that cylinder
    - Then to the rest cylinders towards innermost
- Mapping allows converting a logical block to <cylinder#, track #, sector #>
- In practice, the mapping more complicated
  - Masking defective sectors

Disk Scheduling

- Problem statement:
  - Given the mapping of 1-D array of logical blocks to the sectors of disk, and disk requests keep arriving, schedules disk requests currently in the queue to maximize the disk I/O throughput
  - *Simplification:* the scheduler knows above which cylinder the disk head is, but not which sector
    → Minimize seek time

Disk Scheduling vs. CPU scheduling

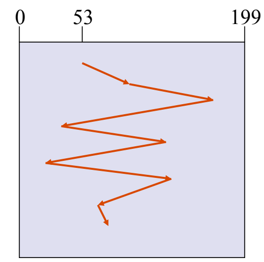
- Similarities:
  - Jobs arrive with uncertainty
  - A set of jobs in the queue to be scheduled
  - Which metrics are similar?
    - throughput, turnaround time, response time, fairness
- Differences?
  - Temporal vs. spatial
  - Scheduling unit

Let us talk about Elevators!

- "The first reference to an elevator is in the works of the Roman architect Vitruvius, who reported that Archimedes (c. 287 BC – c. 212 BC) built his first elevator probably in 236 BC."
- "The first electric elevator was built by Werner von Siemens in 1880."
- How should an elevator decide where to go and stop?

FIFO (FCFS) order

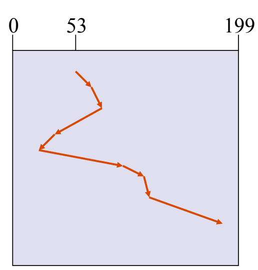
- Method
  - First come first serve
- Pros?
- Cons?



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

SSTF (Shortest Seek Time First)

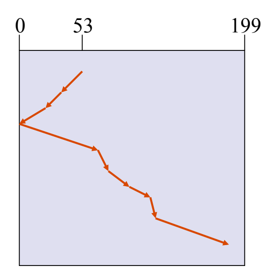
- Method
  - Pick the one closest on disk
- Pros?
- Cons?
- Question
  - Is SSTF optimal? Why?
  - How can we avoid starvation?



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
  - Real implementations do not go to the end (called LOOK)
- Pros?
- Cons?



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

The elevator algorithm

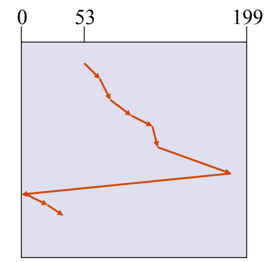
A simple algorithm by which a single elevator can decide where to stop:

- Continue traveling in the same direction while there are remaining requests in that same direction.
- If there are no further requests in that direction, then stop and become idle, or change direction if there are requests in the opposite direction.

Modern elevators use more complex heuristic algorithms to decide which request to service next.

C-SCAN (Circular SCAN)

- Method
  - Like SCAN
  - But, wrap around
  - Real implementation doesn't go to the end (C-LOOK)
- Pros?
- Cons?



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

Which Scheduling Algo to choose?

- SCAN (C-SCAN) good for heavy loads
- If traffic is low, all behave the same as FCFS
- Given list of requests (& their arrive time), optimal solution expensive to calculate
  - Cost may not justify gain over simple solutions
- In general, do not know future

Which Scheduling Algo to choose?

- Future requests depend on
  - File system layout
    - what if inode and data blocks are on the same track
  - File (disk block) allocation method
    - contiguous vs. non-contiguous
- Rule-of-thumb:
  - Decouple disk scheduling from above complications
  - Either SSTF or LOOK a good start

What happens in reality?

- For modern disks, rotational latency as big as seek time
  - Physical location of logical blocks hidden
- Disk manufacturers implement disk scheduling in the controller (e.g. SCSI)
  - Seek time
  - Rotational latency
- OS may still be involved
  - Priority (demanding paging vs. application I/O)
  - Writes more urgent than reads
  - Guarantee order of certain writes (flushing metadata vs. data)

  → OS exploits the freedom to"spoon-feed" disk controller

On-Disk Caching

- Method
  - Disk controller has a piece of RAM to cache recently accessed blocks
    - Seagate Barracutda SATA disk have 32MB
    - Some of the RAM space stores "firmware" (a mini OS)
  - Blocks are replaced usually in LRU order
- Pros
  - Good for reads if you have locality
- Cons
  - Expensive
  - Need to deal with ***reliable*** writes

A startup company idea?

- *Random block access time = seek time + rotational delay + reading time*
- If you have double degrees in EE and ME, can you think of a revolutionary idea?
- Do you think you will get investors excited?

Reading assignment

- Chapter 12 & 13