CS 736: Advanced Operating Systems

Andrea Arpaci-Dusseau

Lecture 3: Background – Disks and File Systems

Papers for Today: Disk Modeling and FFS

Questions for Today:
How do the physical characteristics of disks impact I/O performance?
What techniques did FFS use to improve performance?

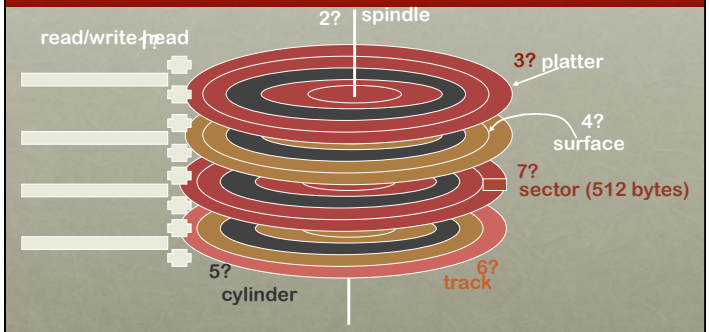
To do for Friday:
Read papers for next lecture:VAX/VMS
Start mini-project (due Sept 27)
Form reading group and email by Friday

What is in a Hard Disk Drive (HDD)?

Electromechanical
- Rotating disks
- Arm assembly

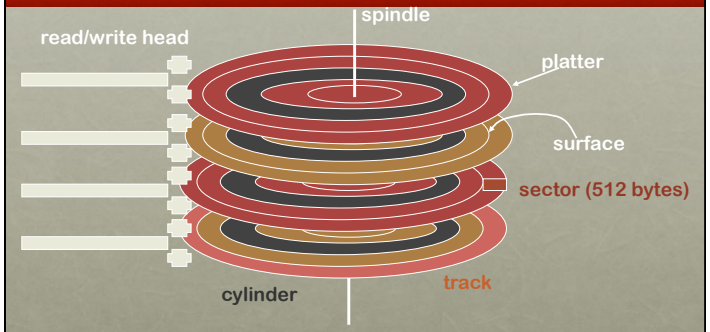
Electronics
- Disk controller
- Cache
- Interface controller

Disk Terminology



Blocks: Group of consecutive sectors
Sector also has overhead information
Error Correcting Codes (ECC)
Servo fields to properly position the head

Disk Terminology



Blocks: Group of consecutive sectors
Sector also has overhead information
Error Correcting Codes (ECC)
Servo fields to properly position the head

HDD Organization

Typical configurations seen in disks today
- Platter diameters: 3.7", 3.3", 2.6; Mobile disks can be as small as 0.75"
- RPMs: 5400, 7200, 10000, 15000
  - 0.5-1% variation in the RPM during operation
- Number of platters: 1-5

Hypothetical disk:
- # surfaces: 4
- # tracks/surface: 16K
- # sectors/track: 256
- # bytes/sector: 512 bytes

What is the capacity?
- 4*16K*256*512 = 8GB

How can you change configuration to reduce power?

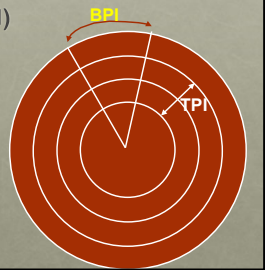
Which has the greatest impact?

Power proportional to: $(\#\ Platters)*(RPM)^{2.8}(Diameter)^{4.6}$
- Tradeoff in the drive-design

Density is increasing!

Density determines capacity and performance

How is density measured?
- Linear density (Bits/inch or BPI)
- Track density (Tracks/inch or TPI)
- Areal Density = BPIxTPI



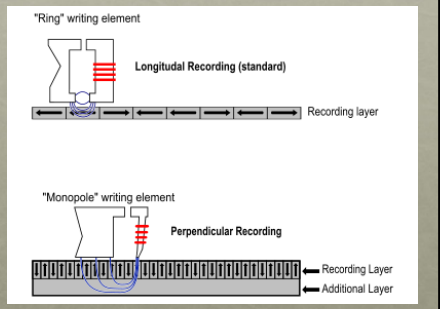
How to Increase Density?

Longitudal Recording

Magnetic domains oriented in the direction in which head travels

Perpendicular Recording

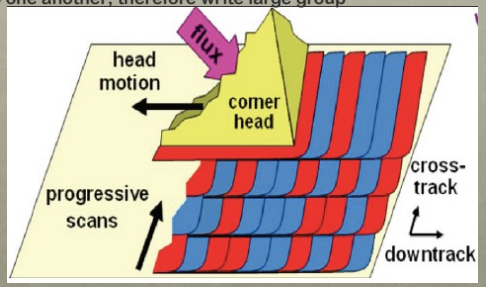
Soft underlayer (mirrors) write field and allows domains to be closer.



Increase Capacity with Shingled Disks

Observation:
Writes are wider than reads and require gap to not interfere

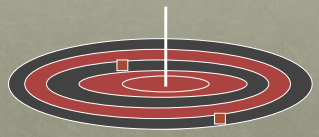
SMR: Shingled Magnetic Recording – Increase density further
Writes overlap one another; therefore write large group



Disk Performance

How long to read or write n sectors?
- Positioning time + Transfer time (n)
- What is positioning time?
  - Positioning time: Seek time + Rotational Delay
  - Seek: Time to position head over destination cylinder
  - Rotation: Wait for sector to rotate underneath head
- What is transfer time?
  - Transfer time: n / (RPM * bytes/track)

Typical access time?
- Order of milliseconds (RAM? Nanoseconds!)



How long to seek?

Simple seek – Linear in distance (max due to platter size)
Seek time depends on:
Inertial power of the arm actuator motor
Distance from outer to inner disk recording track (data-band)

Components of a seek:
- Speedup
  - Arm accelerates
- Coast
  - Arm moving at maximum velocity (long seeks)
- Slowdown
  - Arm brought to rest near desired track
- Settle
  - Head is adjusted to access desired location

Very short seeks (2-4 cylinders)
- Settle-time dominates

Short seeks (200-400 cylinders)
- Speedup/Slowdown-time dominates

Longer seeks
- Coast-time dominates

With smaller platter-sizes and higher TPI?
- Settle-time becoming more important

Switching Time

Head Switch: Switch data channel from one surface to next in same cylinder
- Vertical alignment of cylinders difficult at high TPI
- Head might need to be repositioned during the switch
- Can be 1/3-1/2 of settle-time

Track Switch: Position moves from last track of cylinder to first track of next cylinder
- Almost same amount as the settle-time

At high TPI, head-switching and track-switching times are nearly the same

How to Optimize Performance?

Optimistic Positioning
- Attempt read when head is near the desired track
- ECC and sector ID determine if correct data was read
- Not done for settle before write!

Track Skewing
- To provide faster sequential access across track and cylinder boundaries
- Skew logical sector zero of each track by worst-case head/track switch-time
- Each zone has different skew factors

Data Layout

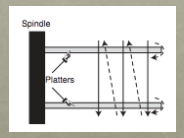
Logical blocks mapped to physical sectors on disk drive

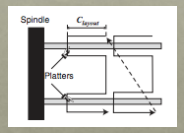
Logical view of disk by OS: Linear array (list) of blocks

D	D	A	A	A	D	B	B	B	B	C	C	C	B	B	D	B	D

Low-Level Layout Factors

- Track Skewing
- Sparing
- Serpentine Layout





- Zoned-Bit Recording (ZBR or multi-zone disk)

Zoned-Bit Recording

Outer tracks can hold more sectors due to larger perimeter

Per-track storage-allocation requires complex channel electronics

Tradeoff:

- Group tracks in zones
- Outer zones allocated more sectors than inner ones
- Due to constant angular velocity, outer zones experience higher data rates.

Modern disks have about 30 zones

On-Disk Cache

Track buffer or larger segmented disk cache

- Typically 8-16 MB for modern disks

Reads

- Sequential reads: Actively reading disk data and placing in cache

Writes

- What is the advantage of buffering writes in disk cache?
  - Provides write-coalescing for better utilization of disk bandwidth
  - Many write requests enables disk scheduling opportunities
  - Time for write to cache is much faster than disk surface
- What is the disadvantage?
  - Reliablity: What if power failure before write to surface?
  - Immediate Reporting: Writes are done as soon as they are written to cache
  - Disable Immediate Reporting if care about durability!

Tagged Command Queuing

Disk can schedule multiple requests itself

- Disks know low-level layout for best ordering

SCAN and elevator

- Handle requests in order by cylinder
- Variation: C-SCAN: Start requests in same dir always

Shortest-seek-time-first (SSTF)

Shortest-positioning-time first (SPTF)

- Take rotational delay into account as well

Drive Electronics

ECC
- Appends ECC symbols, performs error-handling operations
- Current disks employ Reed-Solomon codes

Servo Control
- For necessary signal-processing for disk-rotation and head positioning
- Needed due to motor variation, platter waviness (circumferentially and radially), stacking tolerances, vibrations, etc.

Reliability

Key metric – Mean-Time Between Failures (MTBF)

Typical MTBF for SCSI disk = 1,200,000 hours
- This is typically the first-year reliability
- Assumes "nominal" operating conditions

Factors Affecting Reliability?

Temperature
- Reliability decreases with increase in temperature
- 15 C rise in room temp can double failure-rate

Vibrations
- Caused by moving components nearby
- Cause off-track errors that delay I//O or cause temporary errors
- Worse at high TPI due to smaller tolerances
- Server-disks designed for more vibration tolerance

Disk in Action

FFS Motivation

Original UNIX File system from Bell Labs
- Simple and elegant
- Problem: Achieves 20 Kb/sec
  - 2% of disk maximum even for sequential disk transfers!

Why such poor performance?
- Three primary reasons..

Why such poor performance?

Blocks too small (512 bytes); Two problems?
- Fixed costs per transfer
  - Seek time, rotational delay, computation
- More indirect blocks needed for same size file

Poor freelist organization; Problem?
- Consecutive file blocks not close together
  - Pay seek cost between even sequential disk transfers

No locality in allocation to disk; Two problems?
- I-nodes far from data blocks
  - Pay two seeks for every data transfer
- I-nodes of files in directory not close together
  - Pay seek for every i-node (e.g., ls -l)

#1: Larger Block Sizes

Measure FS performance on workload given different block sizes

Move from 512 bytes to 1KB doubled performance

BSD: Why not increase block to 4096 or 8192 bytes

What is the problem with larger blocks?

What is the solution?

Solution to Internal Fragmentation?

Fragments: Allow large blocks to be chopped into small ones
- Lower bound on size determined disk sector
- Limit number of fragments per block to 2, 4, or 8
- Keep track of free fragments

Beneficial for small files and ends of files

Algorithm for ensuring fragments only used for end of file
- Only allocate fragments from one block per file
- Coallesce blocks of allocated fragments
- Performance problem if file grows a fragment at a time

Advantages
- Greatly reduces amount of wasted space
- Transfer speeds of larger blocks

#2: Unorganized Freelist

Leads to random allocation of sequential files over time

Initial performance good

...but FS are long-lived entities

What are possible solutions?

Fixing the Unorganized Freelist

Periodically compact / defragment disk
- Disadvantage: Disk not accessible during operation

Organize freelist by address; Disadvantages?
- Costly to insert
- Non-trivial to find contiguous free blocks

Bitmap of free blocks
- Solution used in BSD
- Bitmap: 10010000110110101111 1

#3: Locality

Techniques for keeping related items together

Spread unrelated data far apart

What do you need to keep unrelated data apart?
- Requirement: freespace on disk
- Always find free block near desired location
- Rule of thumb: do not 90% utilization

Leaves room for related things to be placed together

What new organization to support locality did BSD introduce?

New organization structure?

Divide disk into cylinder groups
- Set of adjacent cylinders
- Little seek time between cylinders in same group

Each cylinder groups contains?
- Superblock
  - Vary offset within each cylinder group for reliability
- I-nodes
  - Fixed number per cylinder group
- Bitmap of free blocks
- Usage summary for high-level allocation policy
- Data blocks

Goals for Locality

Maintain locality of each file

Maintain locality of files and inodes in a directory

Make room for locality within a directory

How does BSD achieve each of these goals?
- What heuristics does it use when allocating blocks to disk?

Solution to Achieving Locality

Maintain locality of each file
- Allocate runs of blocks within a cylinder group

Maintain locality of files and inodes in a directory
- Keep files in a directory in same cylinder group

Make room for locality within a directory
- Spread out directories among the cylinders groups
  - Greater than average # of free inodes, smallest # of directories
- Switch to a different cylinder group for large files
  - After 48KB and every 1MB thereafter
  - Prevent one file from filling a cylinder group

Layout: Global vs. Local

Decompose allocation into two steps

Global: Heuristics for allocating files+directories to cylinder groups

Pick "optimal" next block for allocation

Local: Handles request for specific block
- If block available, use it
- If not free, check a sequence of alternatives
  1) Next rotational block on same cylinder
  2) A block within cylinder group
  3) Rehash on cylinder group to choose another group
  4) Exhaustive search

Rotationally Optimal Placement

Skip-sector allocation
- Based on CPU and device speed
- Do not allocate contiguous sectors if CPU not fast enough

Problems?
- Cannot achieve full bandwidth from disk
- Timing may be optimal for reads but not writes
- Best allocation changes if move disk between machines

No longer an issue in current systems!

BSD Performance Improvements

Achieve 20-40% of disk bandwidth on large files
- 10x improvement over original Unix file system
- Does not change over lifetime of FS
- Especially good considering skip-sector allocation
  - Could not achieve better than 50% of peak

Better small file performance

Conclusions

General
- Know performance/reliability of underlying components
- Measure your system to improve it!

FFS
- Fragments: Combines benefits of larger and smaller blocks
- Cylinder groups: Structure to aid locality and improve perf
- Locality: Keep inodes and data blocks from same file + dir together

Friday lecture: Memory Background -- VAX VMS
- What are the roles of the pager versus the swapper in VAX VMS? What information is needed by both the pager and the swapper? Is there any information that one must track that the other does not? If so, what?

Friday: Reading Groups Formed- send email with names

Mini-project due: Friday Sept 27th (2 weeks from Fri)