CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

Fall 2021 L23

Mass Storage

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

FAQ

- Indexed allocation: Index blocks includes pointers to file data blocks.
- Inode: contains pointers to data blocks directly or indirectly.
  - All inodes are in the inode table on the disk.
  - Inode number gives the inode address and identifies a file.
- 2^{nd} chance algorithm: If the reference bit is 1, give that page a second chance.
- Can windows mount files from linux and vice versa?
  - Yes. Look up approaches.
- Why use a hard link? Avoid having another copy in another directory.
- Why use a symbolic link? Convenience.
- Average -
  - Average rotational latency – why ½ a rotation time?
  - Average seek time is 1/3 of max seek time. Why?

Notes

- Project reports/slides: Due Th Dec 2, 2021
- Devp TA Demos: Dec 2-8 M-W. Sign-up for 15 min slot. [Videos]
- Research presentations: Dec 2-8 M-W [Videos]
  - Logistics & Details will be on Teams
- Peer reviews due Sat Dec 11, 2021

Final: Comprehensive but mostly from second half

- Sec 001: Tu Dec 14, 6:20-8:20 PM
- Sec 801:
  - Local students with Sec 001 on-campus
  - Non-local: online during a 24 hr time window
- All SDC students (001, 801): at SDC as arranged.
  - Tu Dec 14, 4-8 PM (must stay until 6:20 PM)

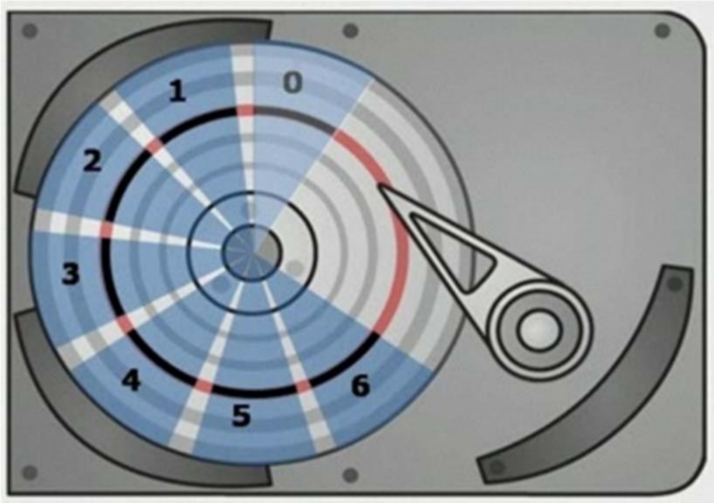
Hard Disk Performance

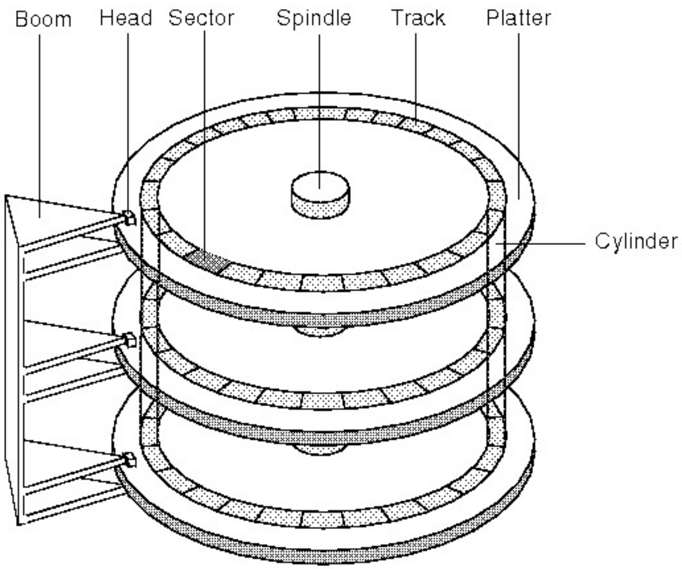
- Average I/O time = average access time + (amount to transfer / transfer rate) + controller overhead
  - **Average access time** = average seek time + average latency
- Example: Find expected I/O time to transfer a 4KB block on a 7200 RPM disk with a 5ms average seek time, 1Gb/sec transfer rate with a 0.1ms controller overhead.
  - Average access time = 5ms + 60/(7200*2) s = 5ms + 4.17ms
  - Transfer time = 4KB / 1Gb/s = 4x8K/G = 0.031 ms
  - Thus **Average I/O time = 9.27ms + .031ms+0.1ms = 9.301ms**

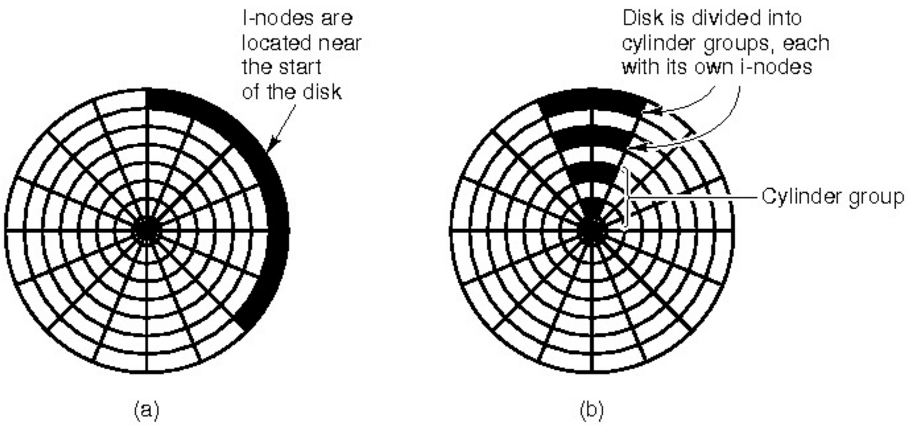
Strategy: memorize formula or understand how it works?

HDD addressing

- Physical: Drive, Cylinder, Head, sector
- Logical Block Addressing (LBA): blocks addressed by numbers.







SSD Architecture

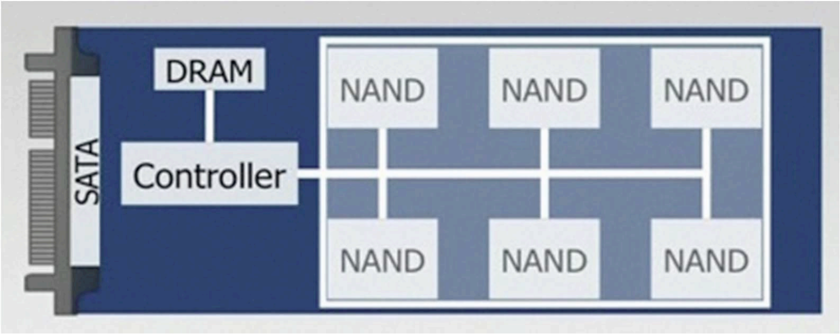
Controller

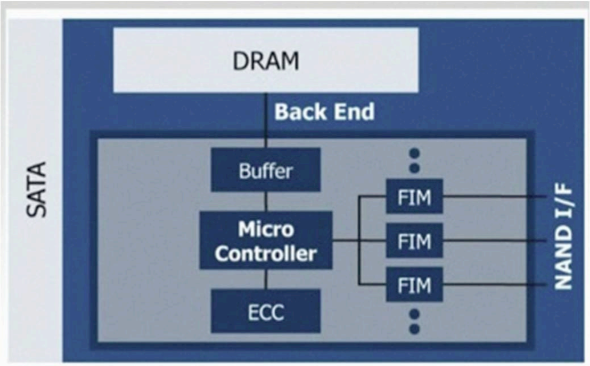
- Takes the raw data storage in the NAND flash and makes it look and act like hard disk drive
- Contains the micro controller, buffer, error correction, and flash interface modules

Micro Controller – a processor inside the controller that takes the incoming data and manipulates it

- Correcting errors
- Manages mapping
- Putting data into the flash or retrieving it from the flash

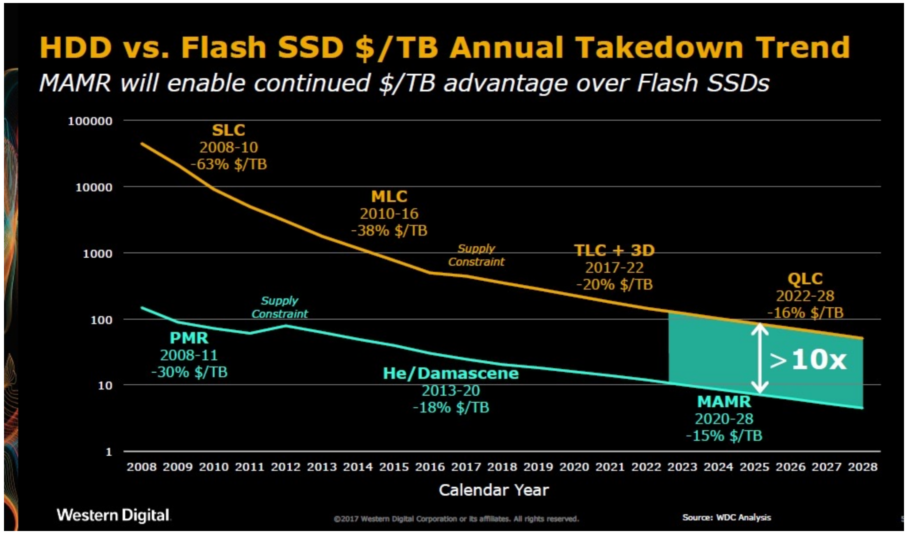
DRAM Cache – Reasonable amount of very low latency

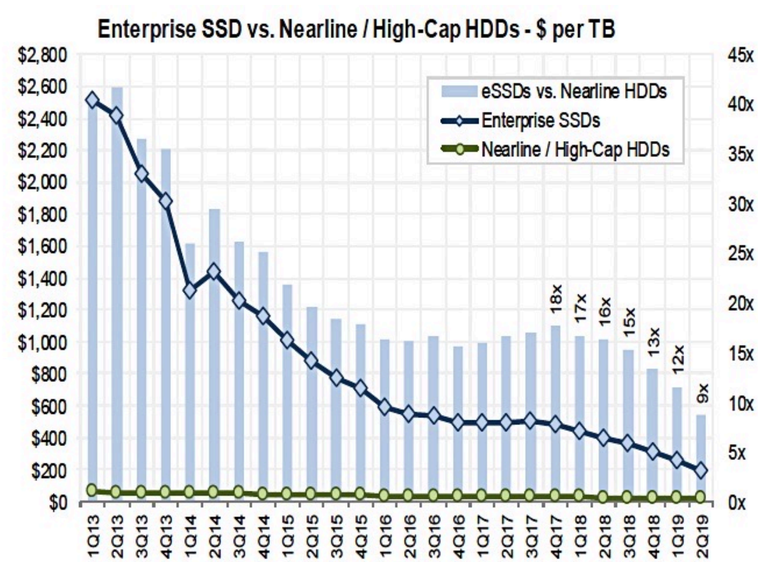




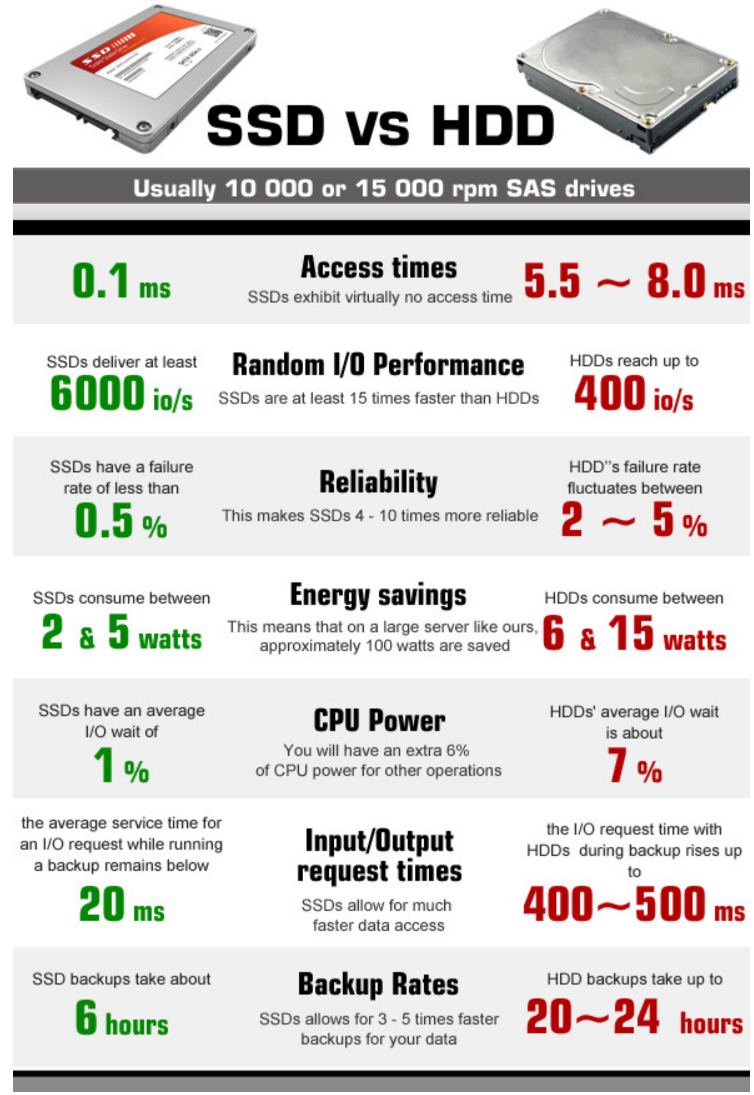
Colorado State University

SSD vs HDD





Source: IDC; TrendFocus; Wells Fargo Securities, LLC



Colorado State University

HDD vs SSD

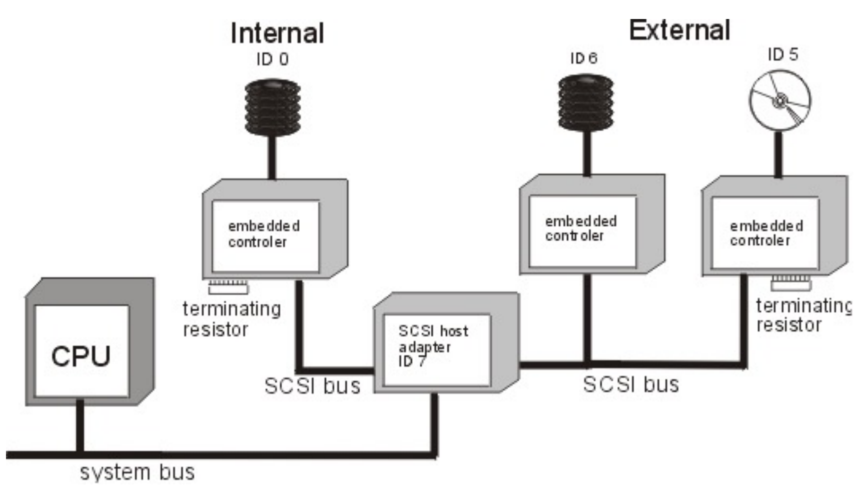
	HDD	SSD
	WD VelociRaptor	OCZ Vertex 3
Storage Capacity	600GB	120GB-360GB
Price for storage	48¢/ GB	2.08$/GB x4
Seek Time/Rotational Speed	7ms/157 MB/s	
MTBF	1.4 million hours?	2 million hours?
Sequential Read/Write	1 MB/s	413.5/371.4 MB/s
Random Read	1 MB/s	68.8 MB/s
Random Write	1 MB/s	332.5 MB/s
IOPS	905	60,000 x60

Magnetic Tape

- Was early secondary-storage medium (now tertiary)
  - Evolved from open spools to cartridges
- Relatively permanent and holds large quantities of data
  - Access time slow
  - Random access ~1000 times slower than disk
  - Once data under head, transfer rates comparable to disk
    - 140MB/sec and greater
- Mainly used for backup, storage of infrequently-used data, transfer medium between systems
- Kept in spool and wound or rewound past read-write head
- 200GB to 1.5TB typical storage Sony: New 330 TB

Disk Attachment: I/O busses

- Host-attached storage accessed through I/O ports talking to I/O busses
- SCSI itself is a bus, up to 16 devices on one cable, **SCSI initiator** (adapter) requests operation and **SCSI targets** (controller) perform tasks
  - Each target can have up to 8 **logical units** (disks attached to device controller)
- FC (fibre channel) is high-speed serial architecture
  - Can be switched fabric with 24-bit address space – the basis of **storage area networks (SANs)** in which many hosts attach to many storage units

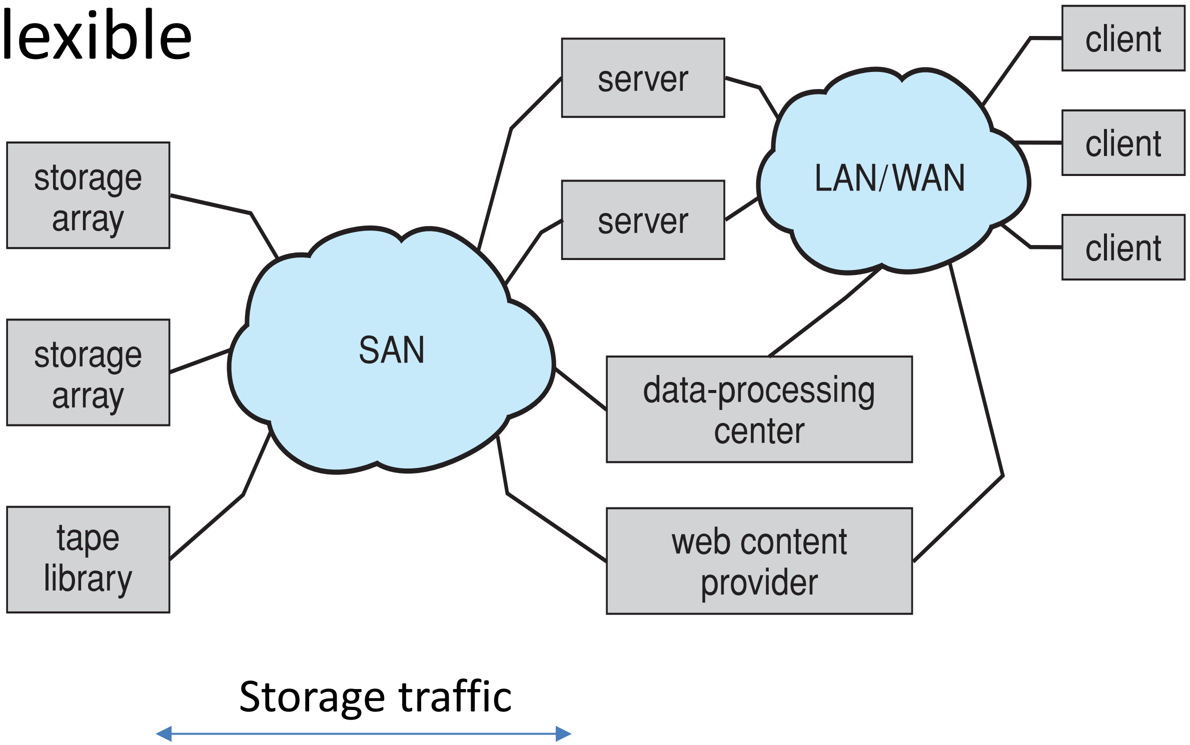


Storage Array

- Can just attach disks, or arrays of disks to an I/O port
- Storage Array has controller(s), provides features to attached host(s)
  - Ports to connect hosts to array
  - Memory, controlling software
  - A few to thousands of disks
  - RAID, hot spares, hot swap
  - Shared storage -> more efficiency

Storage Area Network

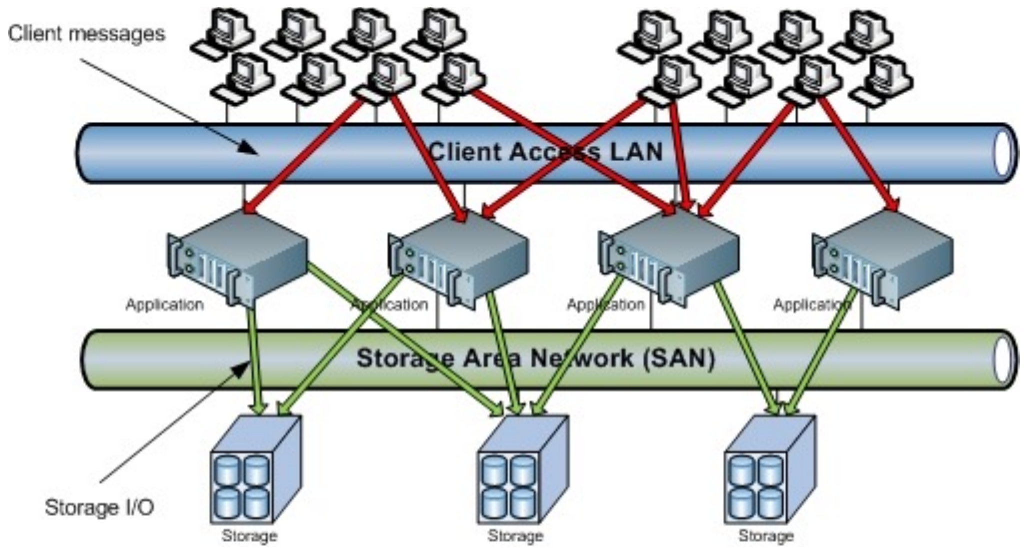
- Common in large storage environments
- Multiple hosts attached to multiple storage arrays - flexible



Colorado State University

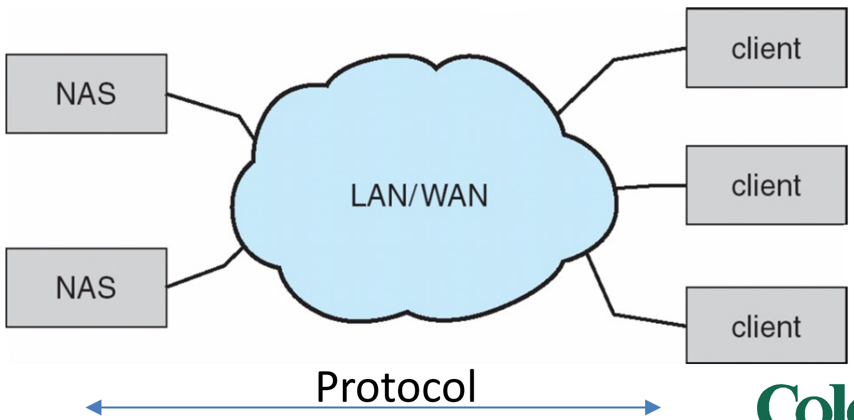
Storage Area Network (Cont.)

- SAN is one or more storage arrays
- Hosts also attach to the switches
- Storage made available from specific arrays to specific servers
- Easy to add or remove storage, add new host and allocate it storage
  - Over low-latency Fibre Channel fabric



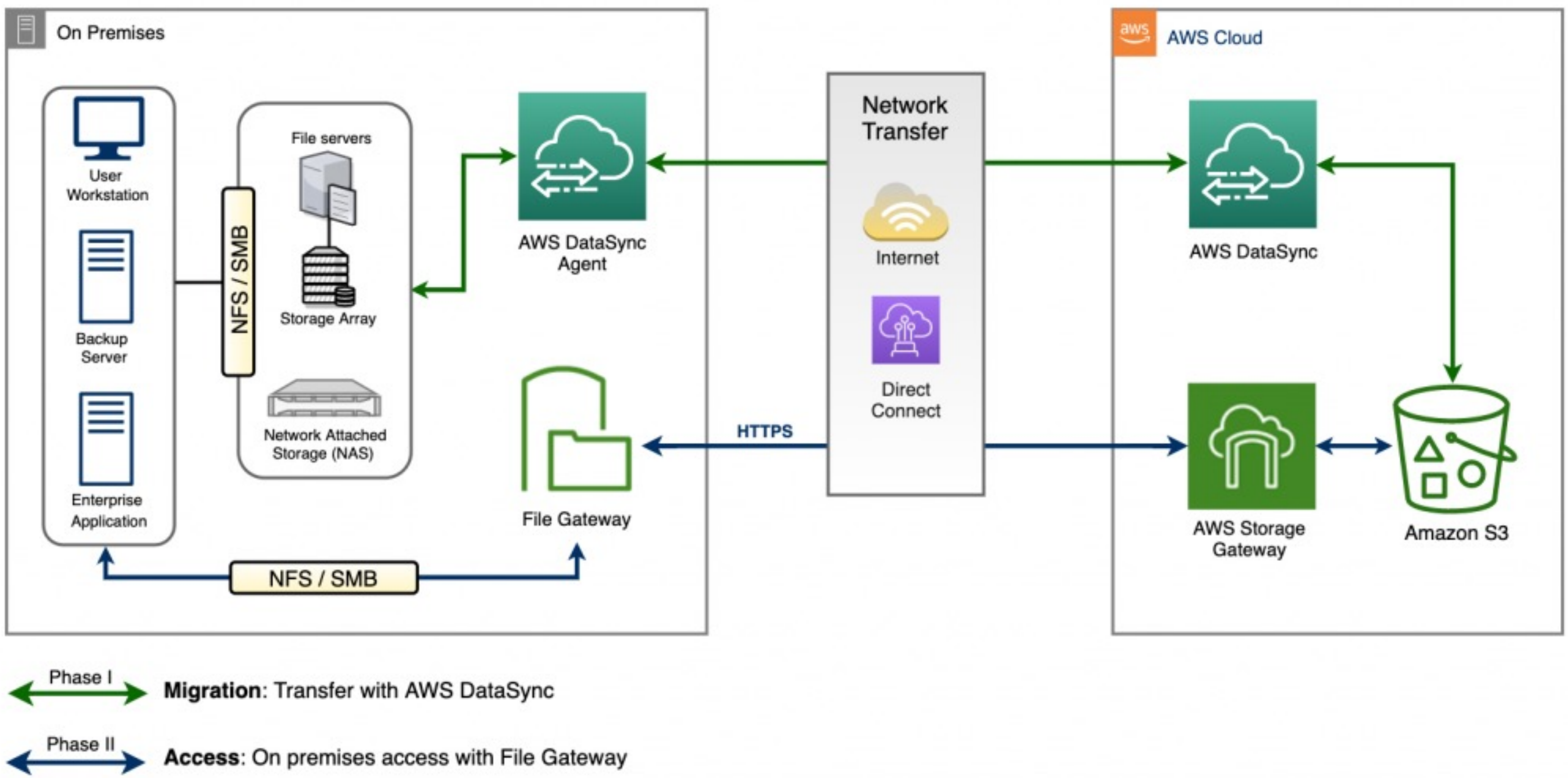
Network-Attached Storage

- Network-attached storage (**NAS**) is storage made available over a network rather than over a local connection (such as a bus)
  - Remotely attaching to file systems
- NFS and CIFS (windows) are common protocols
- Implemented via remote procedure calls (RPCs) between host and storage over typically TCP or UDP on IP network
- **iSCSI** protocol uses IP network to carry the SCSI protocol
  - Remotely attaching to devices (blocks)



Colorado State University

AWS DataSync and Storage Gateway



Amazon S3 (Simple Storage Service)

Issues: Delay, security, availability, cost

https://aws.amazon.com/blogs/storage/from-on-premises-to-aws-hybrid-cloud-architecture-for-network-file-shares/

Disk Scheduling

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth
- Minimize seek time
- Seek time $\approx\propto$ seek distance (between cylinders)
- Disk **bandwidth** is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer

Disk Scheduling (Cont.)

- There are many sources of disk I/O request
  - OS
  - System processes
  - Users processes
- I/O request includes input or output mode, disk address, memory address, number of sectors to transfer
- OS maintains queue of requests, per disk or device
- Idle disk can immediately work on I/O request, busy disk means work must queue
  - Optimization algorithms only make sense when a queue exists

Disk Scheduling (Cont.)

- Note that drive controllers have small buffers and can manage a queue of I/O requests (of varying "depth")
- Several algorithms exist to schedule the servicing of disk I/O requests
- The analysis is true for one or many platters
- We illustrate scheduling algorithms with a request queue (cylinders 0-199)

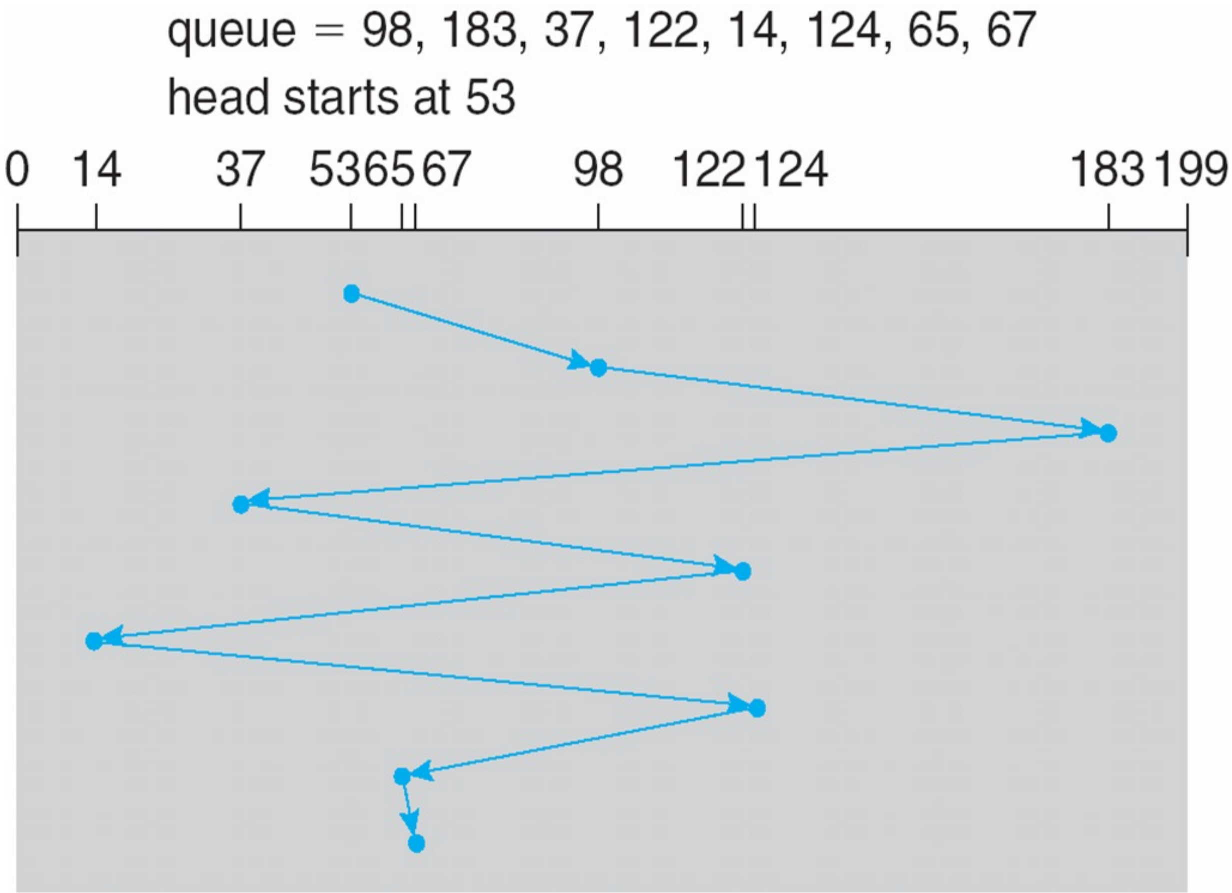
  98, 183, 37, 122, 14, 124, 65, 67

  Head pointer 53 (head is at cylinder 53)

Similar problems: limousine pickup/dropoff, elevator etc.

FCFS (First come first served)

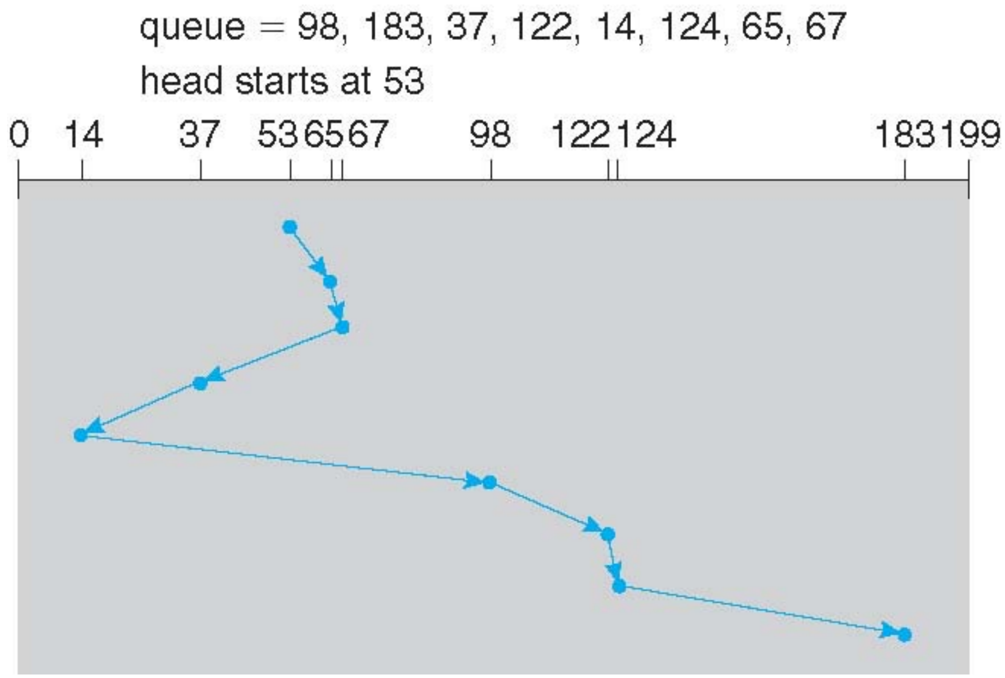
Illustration shows total head movement. Cylinder 0 is outermost



Total seek time = (98-53) + …..= **640** cylinders

SSTF Shortest Seek Time First

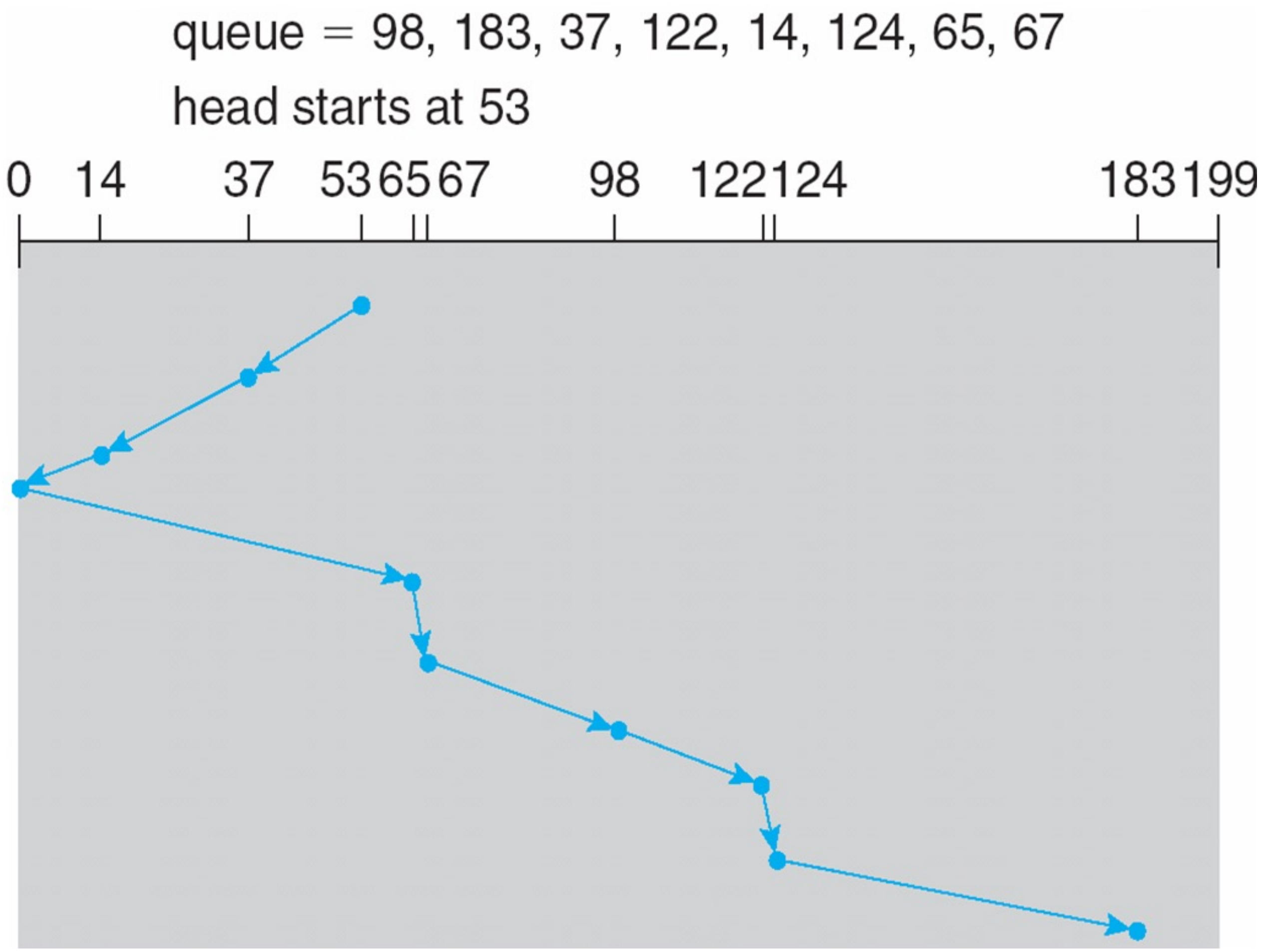
- **Shortest Seek Time First** selects the request with the minimum seek time from the current head position
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests
- total head movement of **236** cylinders



SCAN

- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other **end** of the disk, where the head movement is reversed, and servicing continues.
- **SCAN algorithm** Sometimes called the **elevator algorithm**
- But note that if requests are uniformly dense, largest density at the other end of disk and those wait the longest
- Variation: **Look**: may not go to the very edge

SCAN (Cont.)



LOOK will not go to 0

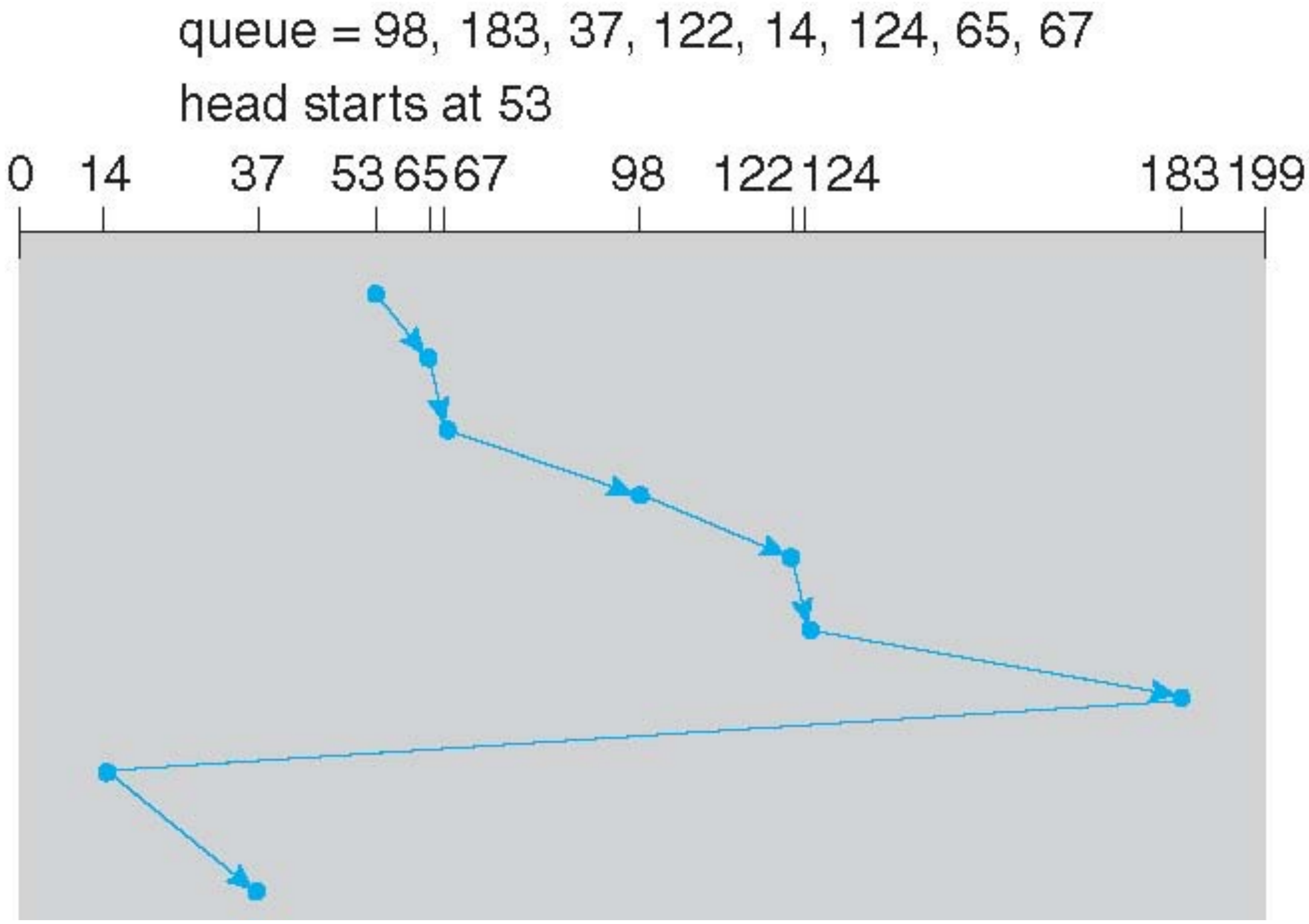
Total 53+ 183= **236** cylinders

Colorado State University

C-LOOK

- LOOK a version of SCAN, C-LOOK a version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk
- Total number of cylinders?

C-LOOK (Cont.)


queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
0
14
37
53
65
67
98
122
124
183
199

Disk Management

- **Low-level formatting**, or **physical formatting** — Dividing a disk into sectors that the disk controller can read and write
  - Each sector can hold header information (sector number), plus data, plus error correction code (**ECC**)
  - Usually 512 bytes of data but can be selectable
- To use a disk to hold files, the operating system still needs to record its own data structures on the disk
  - **Partition** the disk into one or more groups of cylinders, each treated as a logical disk
  - **Logical formatting** or "making a file system"
  - To increase efficiency most file systems group blocks into **clusters**
    - File I/O done in clusters
- Raw disk access for apps that want to do their own block management, keep OS out of the way (databases for example)

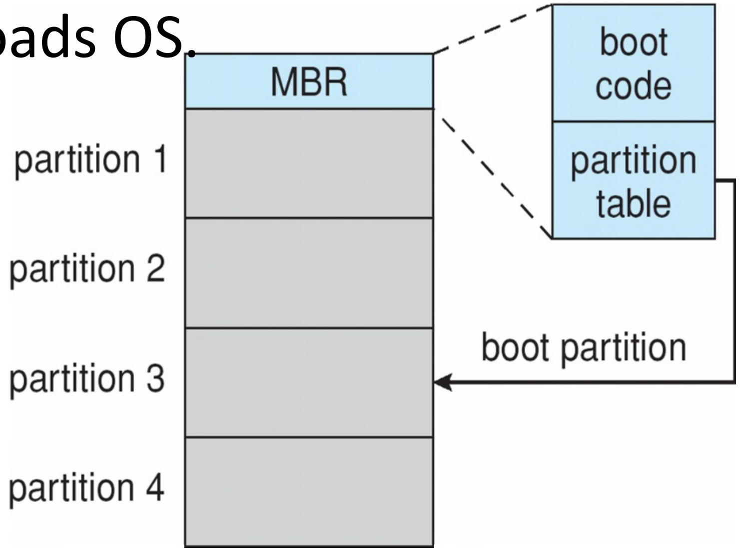
Disk Management (Cont.)

- **Boot block** initializes system
  - The tiny bootstrap code is stored in ROM
  - **Bootstrap loader** program stored in boot blocks of **boot partition which loads OS.**
- Methods such as **sector sparing** used to handle bad blocks

Booting from a Disk in Windows

- **MBR: Master boot record: identifies boot partition**
- **Kernel loaded from boot partition**
- **Boot block** initializes system
  - The tiny bootstrap code is stored in ROM
  - Full **Bootstrap loader** program identified in boot blocks of **boot partition** which loads OS.
- Methods such as **sector sparing** used to handle bad blocks

Boot disk: has a boot partition



CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

Reliability & RAIDs

- Various sources

Reliability

- Storage is inherently unreliable. How can it be made more reliable?
- Redundancy
  - Complete mirrors of data: 2, 3 or more copies.
    - Use a good copy when there is failure,
    - Additional bits: Use parity bit/bits.
    - Use parity to reconstruct corrupted data
  - Rollback and retry
    - Go back to previously saved known good state and re-compute.

RAID Structure

- RAID – redundant array of inexpensive/independent disks. Multiple disk drives provides
  - **Higher reliability, repair capability**
  - **Higher performance /storage capacity**
  - **A combination**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors

RAID Techniques

- **Striping** uses multiple disks in parallel by splitting data: higher performance, no redundancy (ex. RAID 0)
- **Mirroring** keeps duplicate of each disk: higher reliability (ex. RAID 1)
- **Block parity: One Disk hold** parity block for other disks. A failed disk can be rebuilt using parity. Wear leveling if interleaved (RAID 5, double parity RAID 6).
- Ideas that did not work: Bit or byte level level striping (RAID 2, 3) Bit level Coding theory (RAID 2), dedicated parity disk (RAID 4).
- Nested Combinations:
  - RAID 01: Mirror RAID 0
  - RAID 10: Multiple RAID 1, striping
  - RAID 50: Multiple RAID 5, striping
  - others

RAID

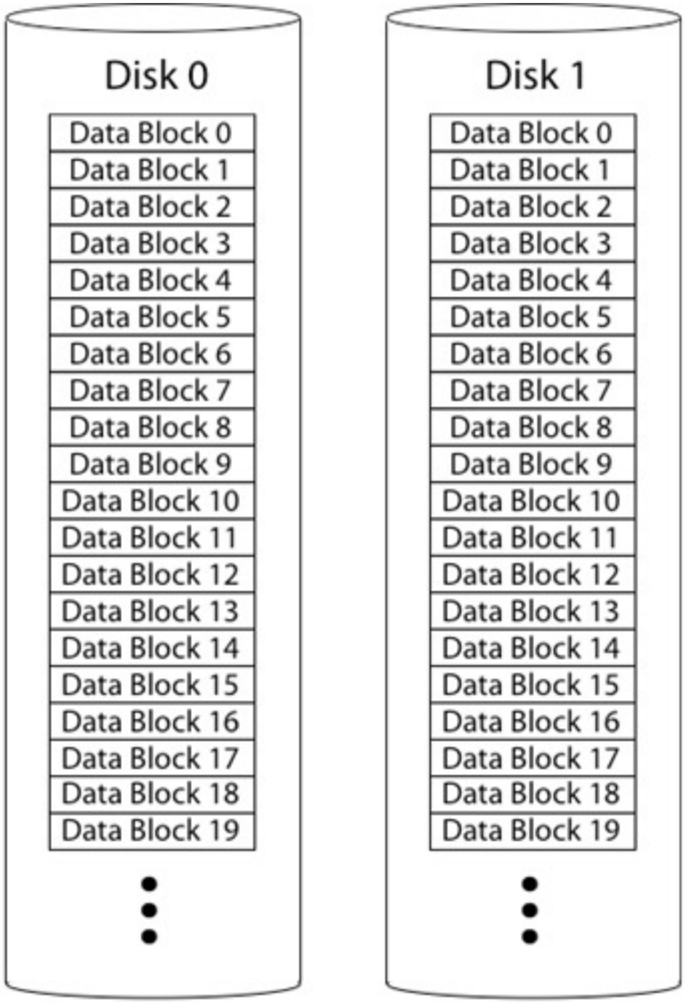
- Replicate data for availability
  - RAID 0: no replication
  - RAID 1: mirror data across two or more disks
    - Google File System replicated its data on three disks, spread across multiple racks
  - RAID 5: split data across disks, with redundancy to recover from a single disk failure
  - RAID 6: RAID 5, with extra redundancy to recover from two disk failures

RAID 1: Mirroring

- Replicate writes to both disks
- Reads can go to either disk
- If they fail independently, consider disk with 100,000 hour *mean time to failure* and 10 hour *mean time to repair*
- One disk fails wile other is being repaired: data loss
  - probability that two will fail within 10 hours =

    $$(2x10)\ /100,000^2$$

  - *Mean time to data loss* is $100,000^2/(2x10) = 500x10^6$ hours, or 57,000 years!



Parity

- Data blocks: Block1, block2, block3, ...
- Parity block: Block1 xor block2 xor block3 ...

10001101	block1
01101100	block2
11000110	block3

00100111	parity block (*ensures even number of 1s*)

- Can reconstruct any missing block from the others

Parity Exercise

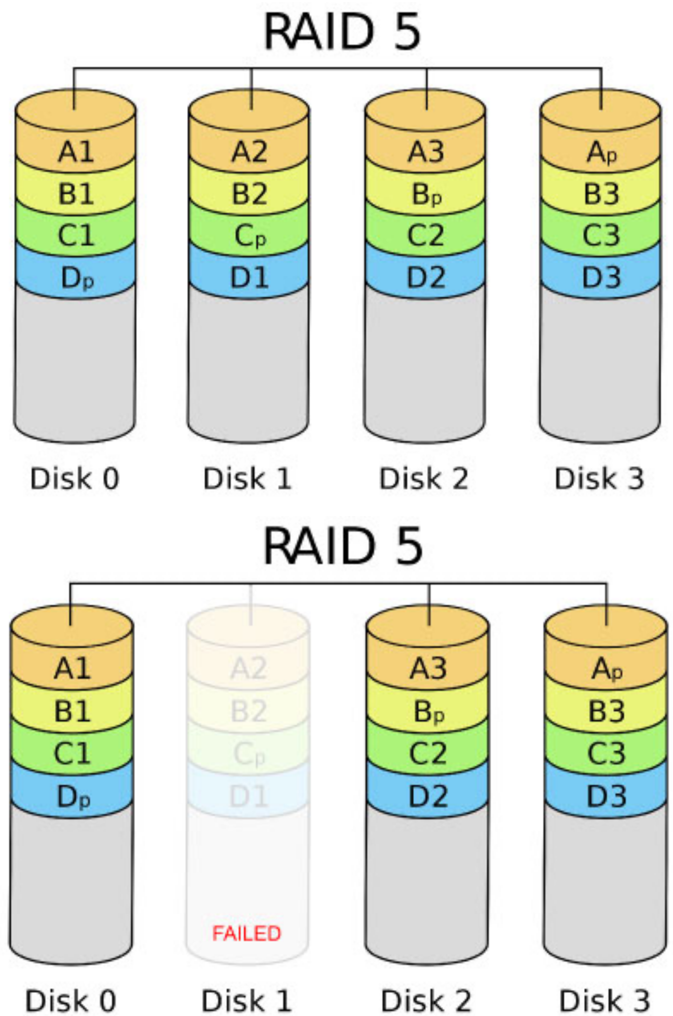
- Parity block: Block1 xor block2 xor block3 …

10001101	block1
????????	block2 (bad)
11000110	block3

00100111	parity block (*ensures even number of 1s*)

- Can you reconstruct the bad block using the others?

RAID 5: Rotating Parity



Time to rebuild depends on disk capacity and data transfer rate

Read Errors and RAID recovery

- Example: RAID 5
  - Each bit has 10^{-15} probability of being bad.
  - 10 one-TB disks, and 1 fails
  - Read remaining disks to reconstruct missing data
- Probability of an error in reading 9 TB disks during recovery

  $= 10^{-15} * (9 \text{ disks} * 8 \text{ bits} * 10^{12} \text{ bytes/disk})$

  = 7.2%. Thus recovery probability = 92.8%
- Even better:
  - RAID-6: two redundant disk blocks
  - Can work even in presence of one bad disk
  - Scrubbing: read disk sectors in background to find and fix latent errors

CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

Big Data: Hadoop HDFS and map-reduce

- Various sources

Hadoop: Distributed Framework for Big Data

Big Data attributes:

- Large volume: TB -> PB varies with Kryder's law: disk density doubles / 13 months
- Geographically Distributed: minimize data movement
- Needs: reliability, analytic approaches

History:

- Google file system 2003 and Map Reduce 2004 programming lang
- Hadoop to support distribution for the Yahoo search engine project '05, given to Apache Software Foundation '06
- Hadoop ecosystem evolves with Yarn '13 resource management, Pig '10 scripting, Spark '14 distributed computing engine. etc.

- *The Google file system* by Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung (2003)
- *MapReduce: Simplified Data Processing on Large Clusters.* by Jeffrey Dean and Sanjay Ghemawat (2004)

Hadoop: Distributed Framework for Big Data

Recent development.

- Big data: multi-terabyte or more data for an app
- Distributed file system
  - Reliability through replication (Fault tolerance)
- Distributed execution
  - Parallel execution for higher performance

Hadoop: Core components

- Hadoop (originally): MapReduce + HDFS
- MapReduce: A programming framework for processing parallelizable problems across huge datasets using a large number of commodity machines.
- HDFS: A **d**istributed **f**ile **s**ystem designed to efficiently allocate data across multiple commodity machines, and provide self-healing functions when some of them go down

- Commodity machines: lower performance per machine, lower cost, perhaps lower reliability compared with special high performance machines.

Challenges in Distributed Big Data

Common Challenges in Distributed Systems

- Node Failure: Individual computer nodes may overheat, crash, have hard drive failures, or run out of memory or disk space.
- Network issues: Congestion/delays (large data volumes), Communication Failures.
- Bad data: Data may be corrupted, or maliciously or improperly transmitted.
- Other issues: Multiple versions of client software may use slightly different protocols from one another.
- Security

HDFS Architecture

Hadoop Distributed File System (HDFS):

- HDFS Block size: 64-128 MB ext4: 4KB
- HDFS file size: "Big"
- Single HDFS FS cluster can span many nodes possibly geographically distributed. datacenters-racks-blades
- Node: system with CPU and memory

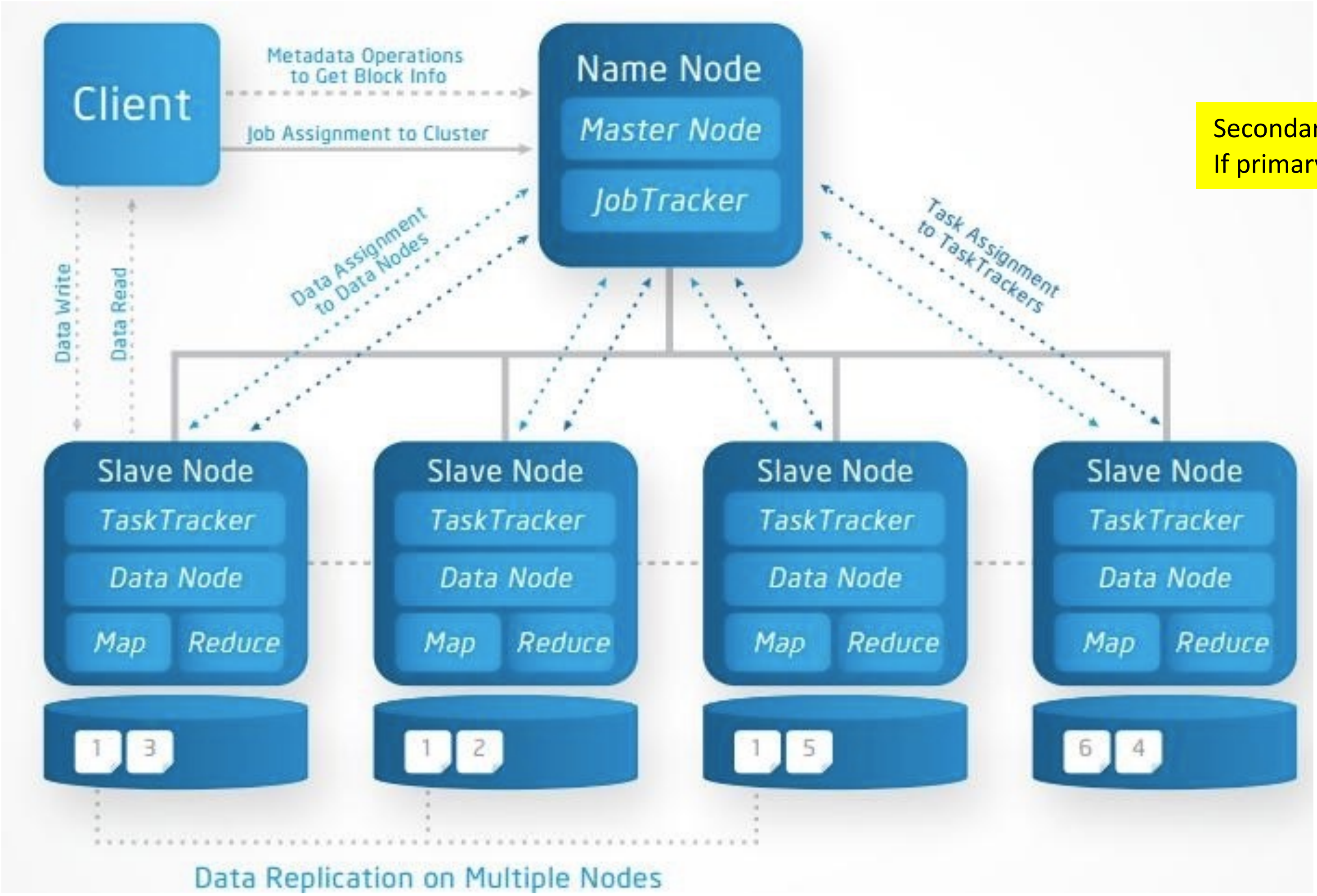
Metadata (corresponding to superblocks, Inodes)

- Name Node: metadata giving where blocks are physically located

Data (files blocks)

- Data Nodes: hold blocks of files (files are distributed)

HDFS Architecture



Secondary Name node
If primary fails

http://a4academics.com/images/hadoop/Hadoop-Architecture-Read-Write.jpg