CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Fall 2024 L23

Mass Storage

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

Hard and symbolic links

Hard Links:

- Both file names refer to the same inode (and hence same file)
  - Directory entry in /dirA

    ..[12345 filename1]..
  - Directory entry in /dirB

    ..[12345 filename2]..
- To create a hard link

  ln /dirA/filename1 /dirB/filename2
- Symbolic link *shortcut* in windows
  - To create a symbolic link

    ln -s /dirA/filenmame1 /dirB/filename3

    File filename3 just contains a pointer

CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

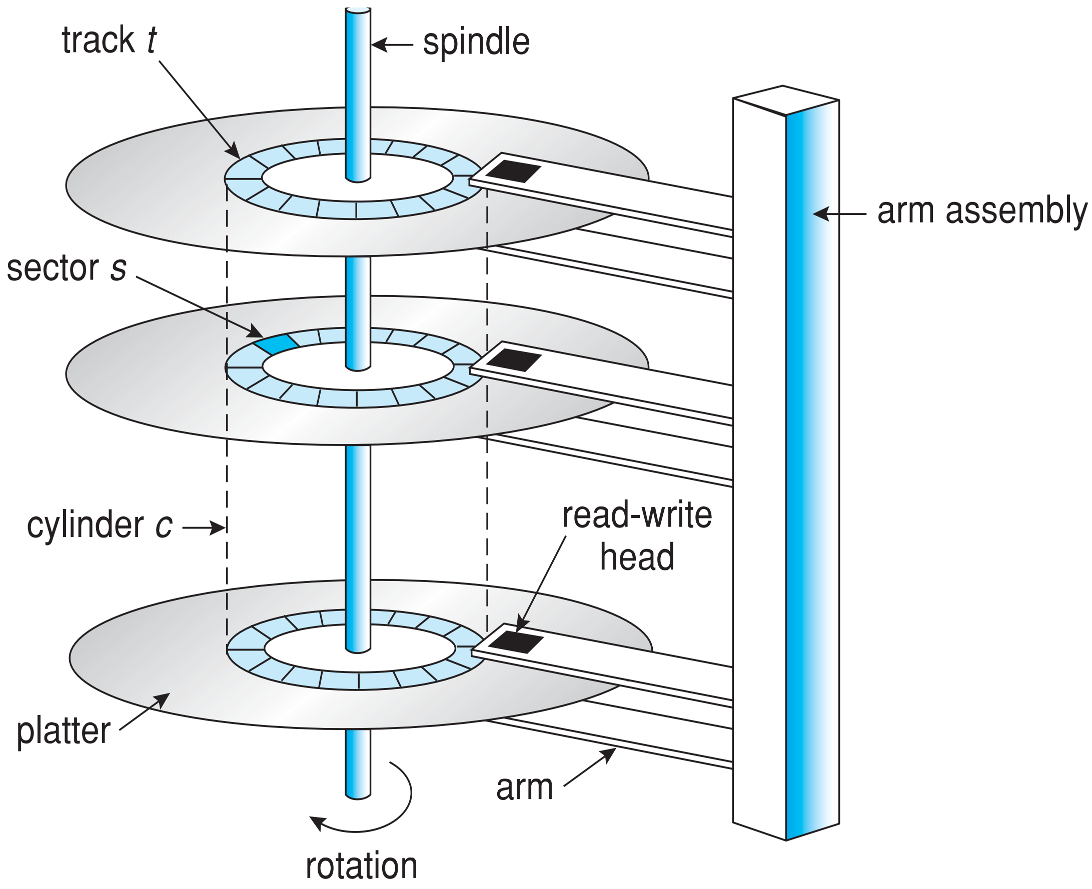
Fall 2022. Ch 11

Mass Storage

Slides based on

- Text by Silberschatz, Galvin, Gagne
- Various sources

Moving-head Disk Mechanism



Hard Disk Performance

- **Average access time** = average seek time + average latency
  - For fastest disk 3ms + 2ms = 5ms
  - For slow disk 9ms + 5.56ms = 14.56ms
- Average I/O time = average access time + (amount to transfer / transfer rate) + controller overhead
- 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.

Av latency =60/(7200*2)

= (5ms + 4.17ms) + 0.1ms + transfer time

- Transfer time = 4KB / 1Gb/s = 4x8K/G = 0.031 ms
- Average I/O time for 4KB block = 9.27ms + .031ms = 9.301ms

Strategy: memorize formula or understand how it works?

Research Project

Objective: Explore state of technology, recent developments, current trends in a field

You need to put effort into

- Digging out the information from news reports, industrial articles/publications, research articles etc. All sources need to be properly cited.
- Connecting the information found and preparing a coherent, well focused report. Non-text information needed: Diagrams, plots, data, tables, flow-charts etc. Cite the sources.
- Readers (students/TAs/Prof) should find the presentation and report interesting and informative.

Use of Generative AI

- Emergence of generative AI is an exciting development. That has created a challenge in academics.
- Use of AI (or copying-and-pasting text) is not permitted in CS370. You must do your own research and write/organize your own report.
- We will check using automated and manual approaches and act as needed.
- A few students have expressed their concern about people in their team using AI generated text, since the responsibility is collective.
- Send me any thoughts privately.

Final Report History

Must be in MS Word (not pdf), and we should be able to see the change history. It must be created using either

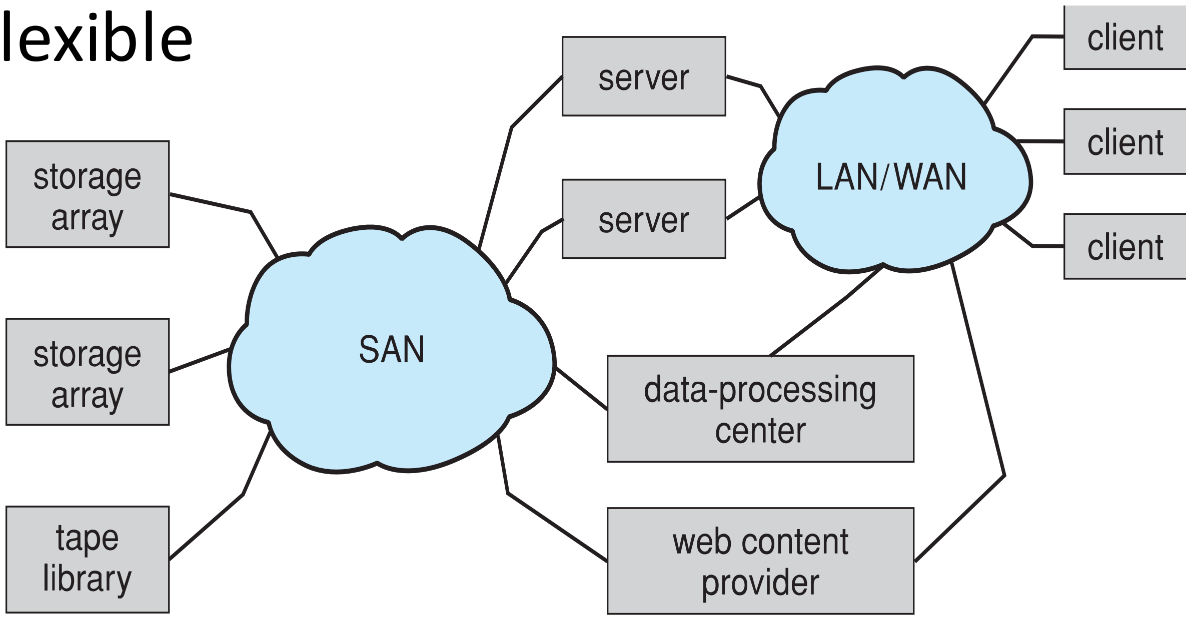
- MS Word with change tracking enabled at the beginning so that we can see the change history.
- Google Doc with a link in the submissions to the Google Doc so that we can view the editing history.

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

Storage Area Network

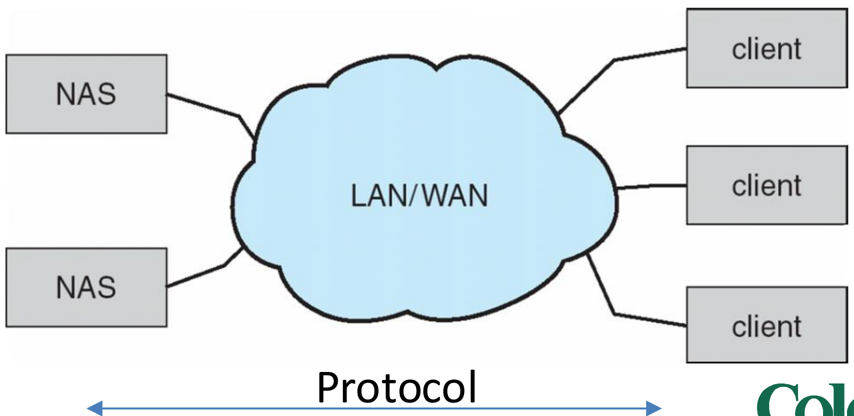
- Common in large storage environments
- Multiple hosts attached to multiple storage arrays - flexible



Storage traffic

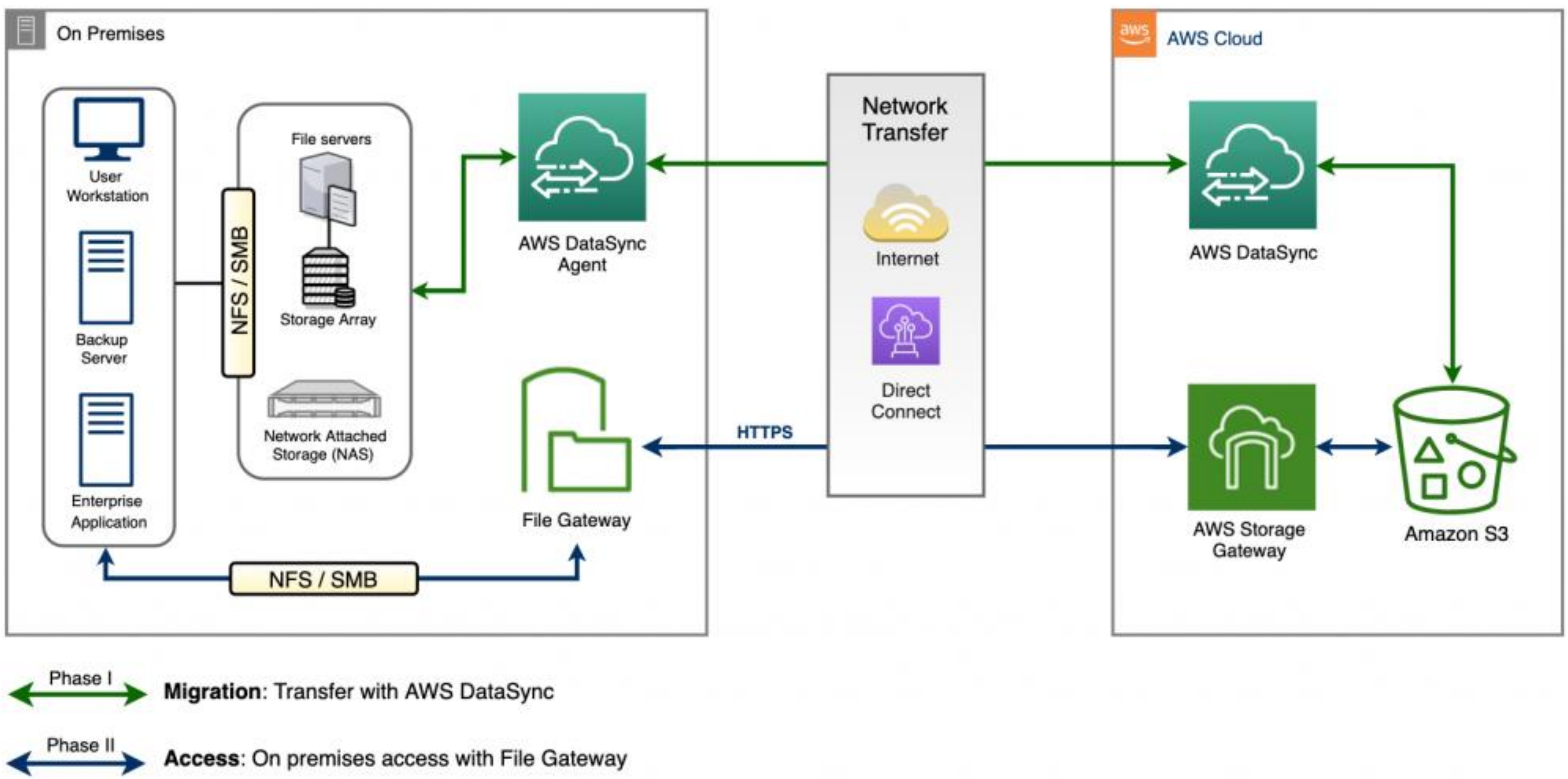
Network-Attached Storage

- 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)



Cloud Storage

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.)

- 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)

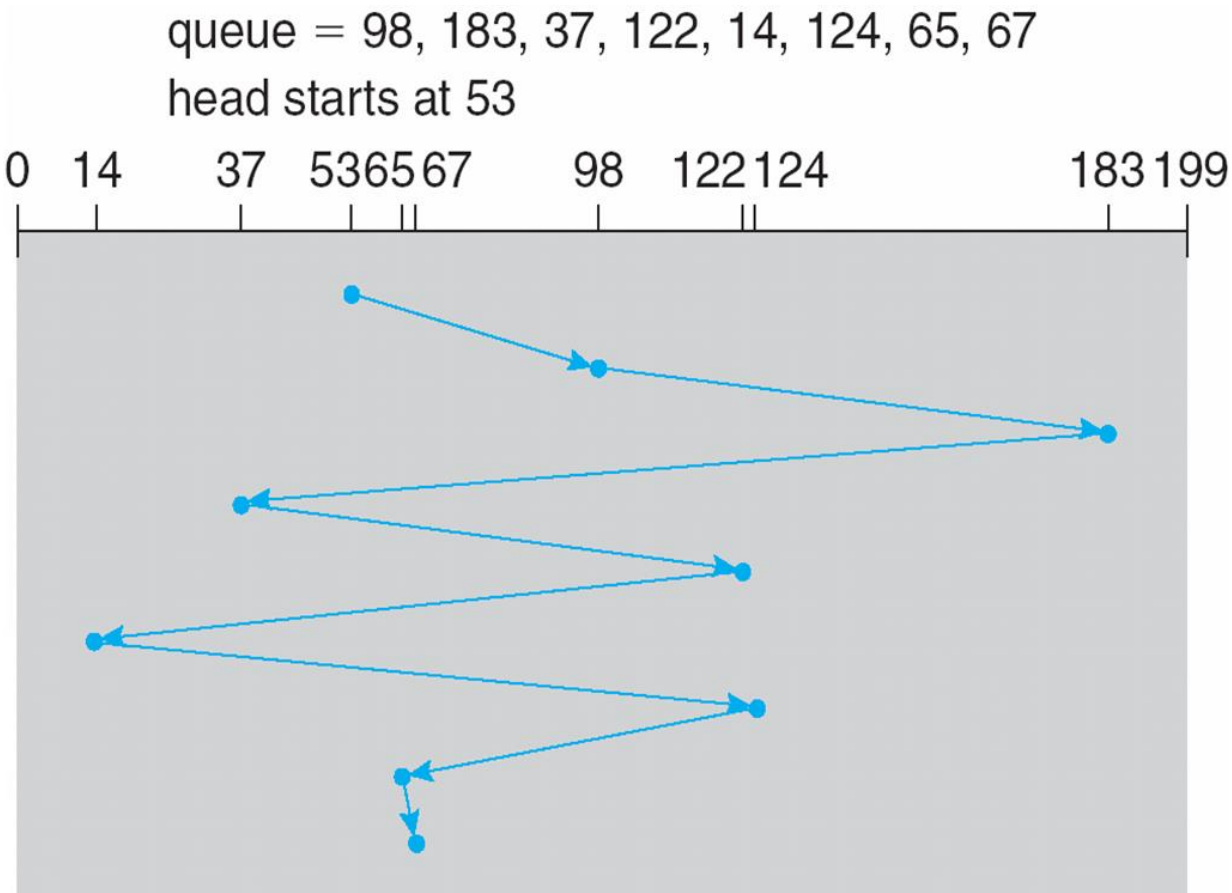
  98, 183, 37, 122, 14, 124, 65, 67

  Head pointer 53 (head is at cylinder 53)

Similar problems: limousine pickup/dropoff, elevator etc.

FCFS (First come first served)

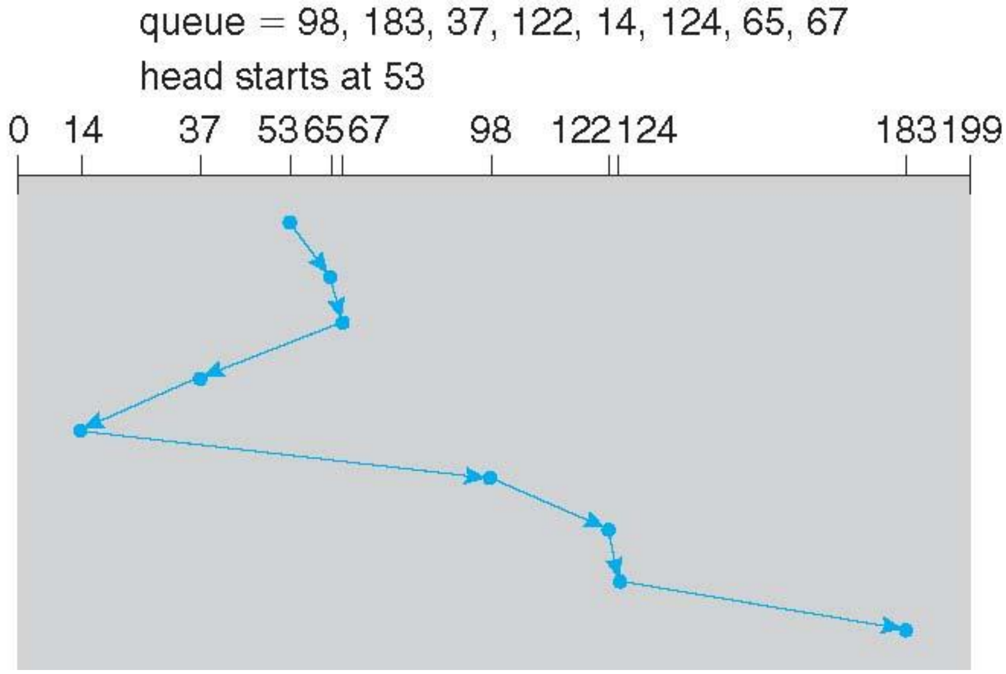
Illustration shows total head movement. Cylinder 0 is outermost



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

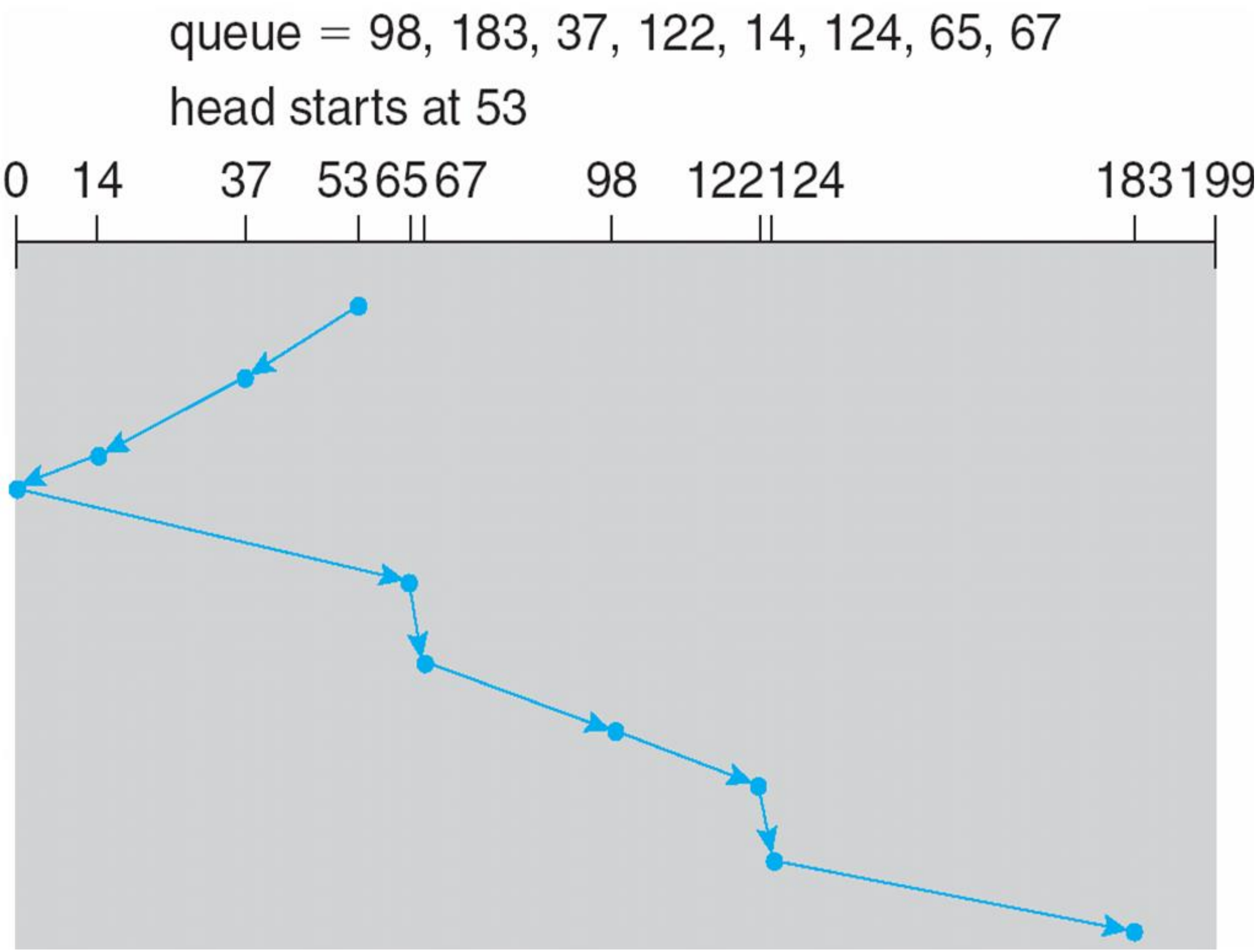


Colorado State University

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.)



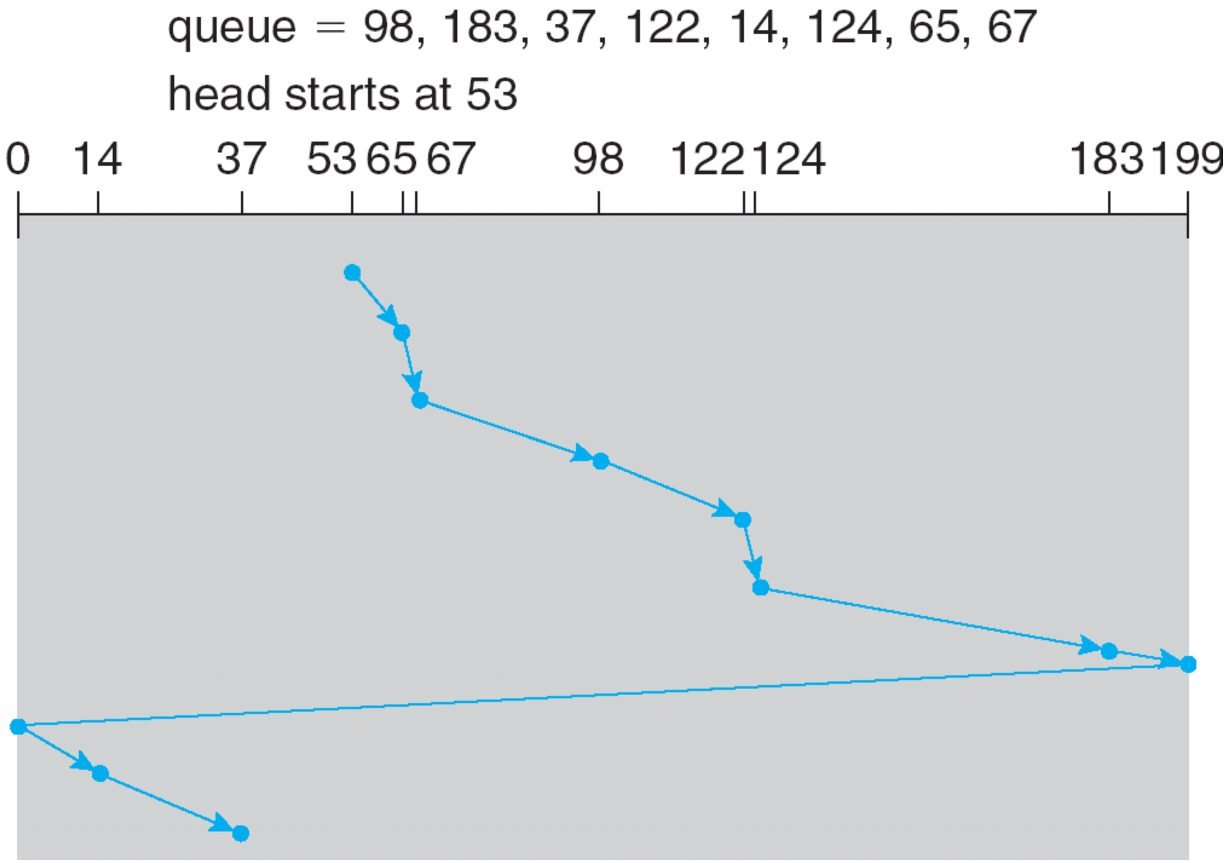
LOOK will not go to 0

Total 53+ 183= **236** cylinders

C-SCAN

- Provides a more uniform wait time than SCAN
- The head moves from one end of the disk to the other, servicing requests as it goes
  - When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one
- Total number of cylinders?

C-SCAN (Cont.)

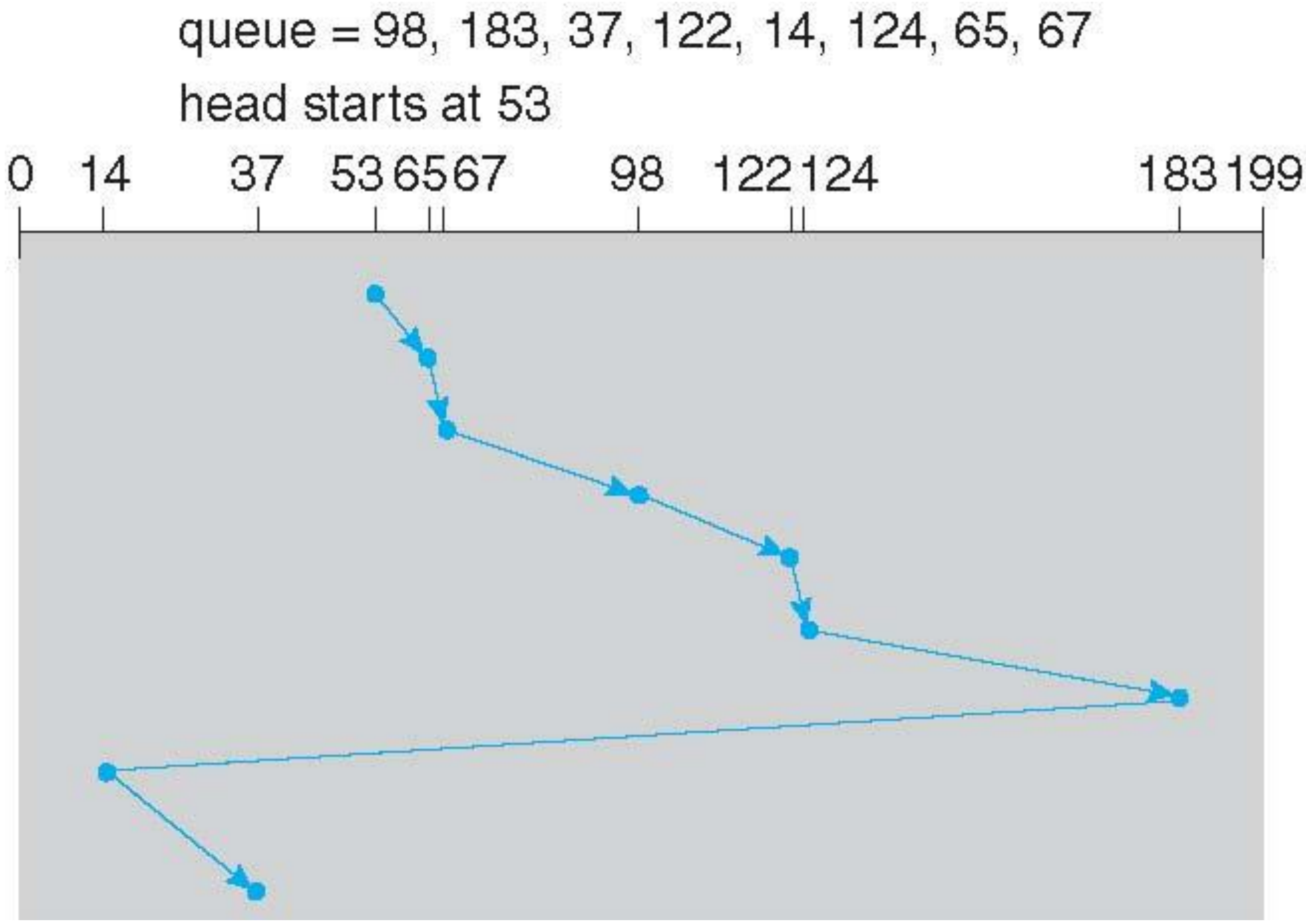


Colorado State University

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

Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk
  - Less starvation
- Performance depends on the number and types of requests
- Requests for disk service can be influenced by the file-allocation method
  - And metadata layout
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary
- Either SSTF or LOOK is a reasonable choice for the default algorithm
- What about rotational latency?
  - Difficult for OS to calculate
- How does disk-based queueing effect OS queue ordering efforts?

CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Reliability & RAIDs

- Various sources

RAID Techniques

- **Striping** uses multiple disks in parallel by splitting data: higher performance (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 (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

Ch 11 + external

Colorado State University

RAID Structure

- RAID – redundant array of inexpensive disks
  - multiple disk drives provides reliability via **redundancy**
  - can increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss.
  - Can be many hours based on size of the disk.
- **Mean time to data loss** based on above factors. Date is lost if an additional failure makes it impossible to restore the data.

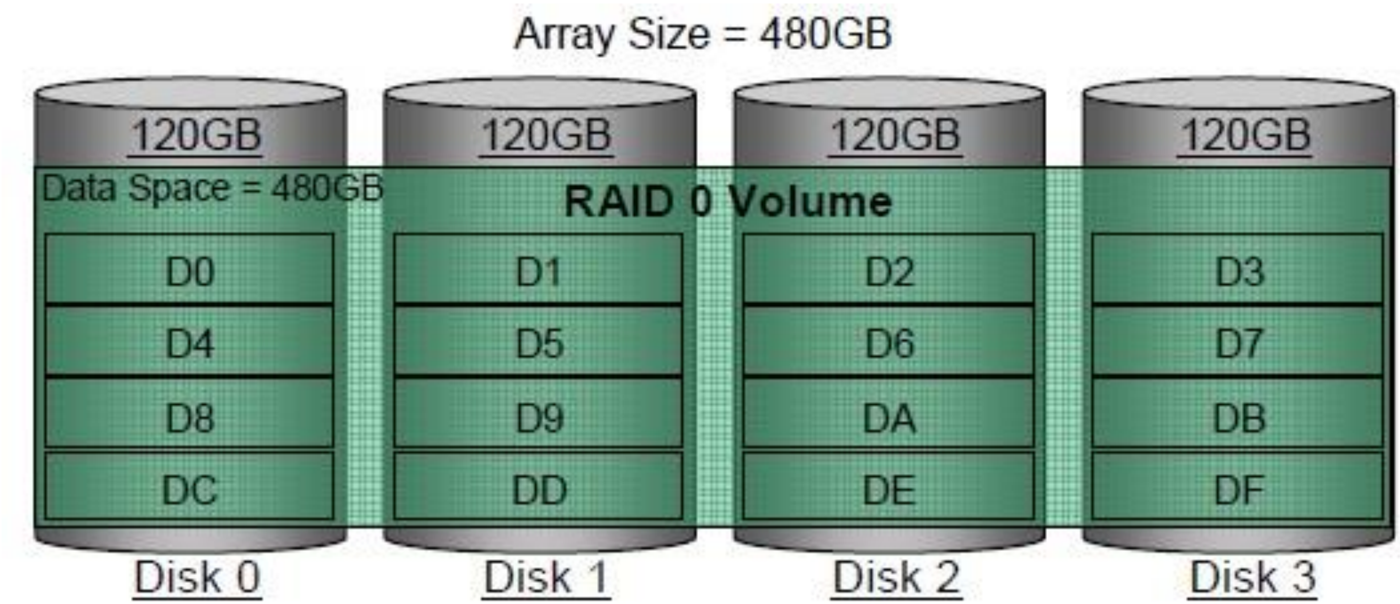
RAID

- Replicate data for availability
  - RAID 0: no replication, data split across disks
  - 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

Failures and repairs

- If a disk has *mean time to failure (MTTF) of* 100,000 hour.
  - Failure rate is 1/100,000 per hour.
- May be estimated using historical data
- If a disk has a bad data, it may be repaired
  - Copy data from a backup
  - Reconstruct data using available data and some invariant property.
- If data cannot be repaired, it is lost.

RAID 0: Striping



- Additional disks provide additional storage
- No redundancy

Colorado State University

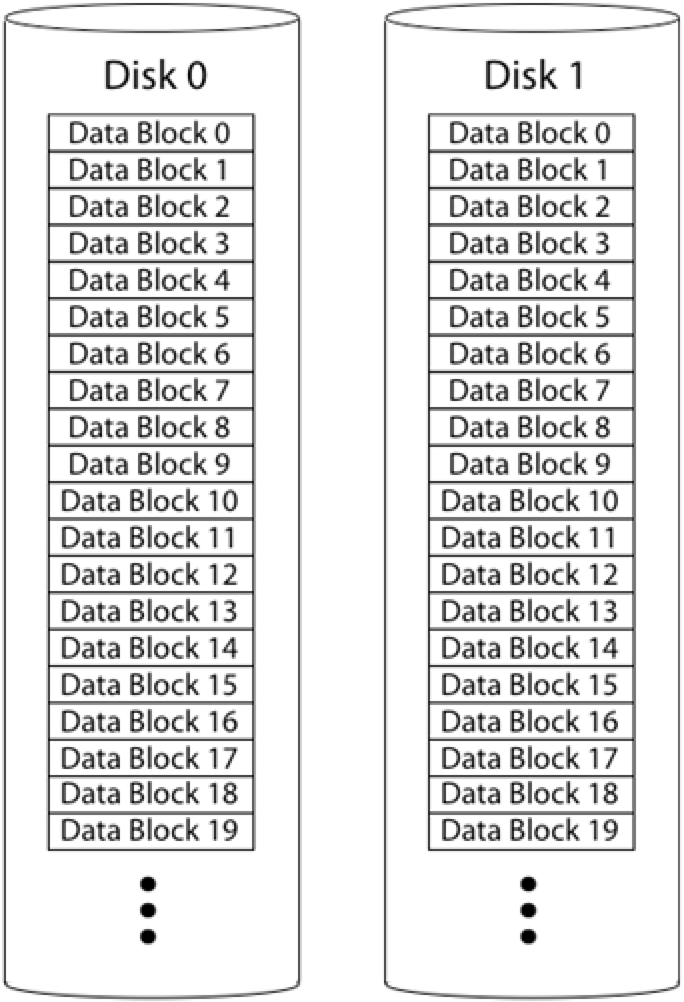
RAID 1: Mirroring

- 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*
  - probability that two will fail within 10 hours =

    $(2 \times 10) \quad / 100{,}000^2$
  - *Mean time to data loss* is

    $100{,}000^2 / (2 \times 10) = 500 \times 10^6$

    hours, or 57,000 years!



Parity bit, Parity block

- **Parity bit(s):** Extra bits obtained using data bits. Used for error detection/correction.
- **Ex:** Parity bit_i = $word_0$ $bit_i \oplus .. \oplus$ $word_n$ bit_i

  = bit needed make 1's even

  – Block parity: bit-by-bit parity for all disks
  – RAID 4: extra disk to hold parity blocks (not used anymore)
  – RAID 5: Parity blocks are distributed among the disks
  – RAID 6: Double the number of parity blocks

Parity

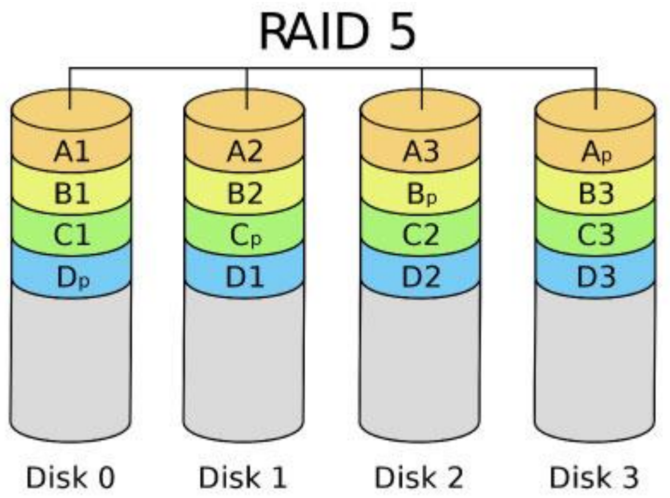
- Data blocks: Block1, block2, block3, ...
- Parity block: Block1 xor block2 xor block3 …

10001101	block1
01101100	block2
11000110	block3

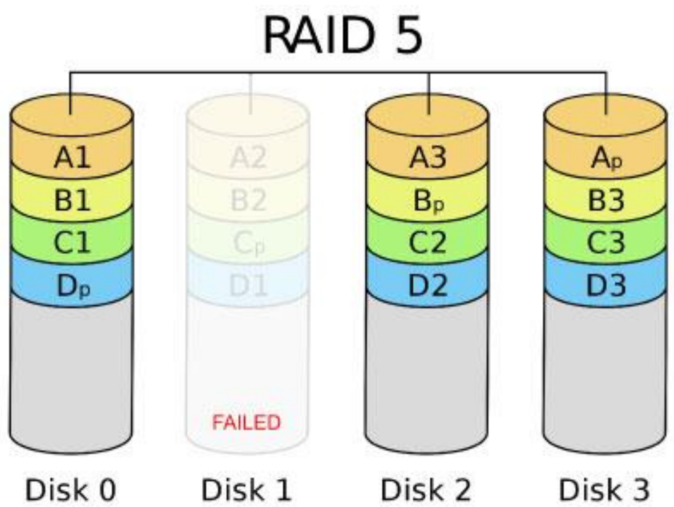
00100111	parity block (*ensures even number of 1s*)

- Can reconstruct any missing block from the others Error-control coding identifies that a block is bad.

RAID 5: Rotating Parity



Parity blocks Ap, Bp, Cp, Dp distributed across disks.



Time to rebuild depends on disk capacity and data transfer rate

Parity bit, Parity block

- RAID recovery:
  - RAID 1: Copy info from good mirror
  - RAID 5,6: rebuild using available data, parity info
- How do we know a disk is corrupted? Use of CRC redundancy at a lower level.

Read Errors and RAID recovery

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

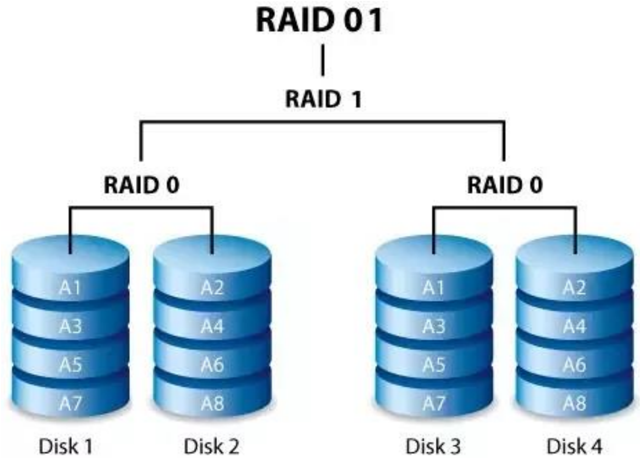
  10^{-15}*total bits =10^{-15}* (9 disks * 8 bits * 10^{12} bytes/disk)

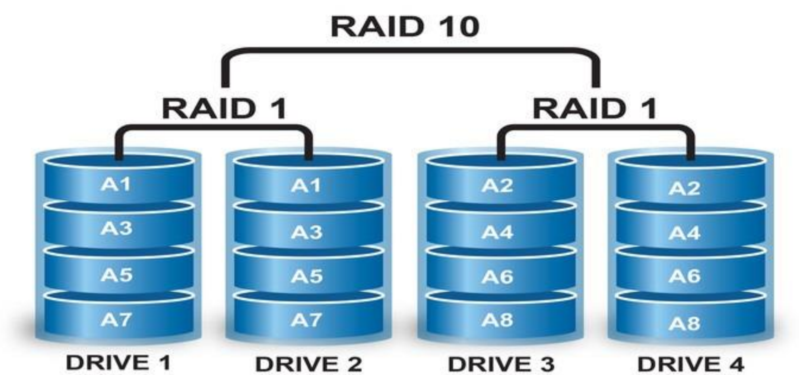
  = 7.2% Thus recovery probability = 92.8%
- Even better:
  - RAID-6: two redundant disk blocks parity plus Reed-Solomon code
  - Can work even in presence of one bad disk, can recover from 2 disk failures
  - Scrubbing: read disk sectors in background to find and fix latent errors

RAIDs: Nested systems

Nested systems: combine striping with mirroring/parity

- RAID 01: Two RAID 0 systems (with striping) mirrored
- RAID 10: Multiple RAID 1 systems (with mirroring) striped.





CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

Big Data: HDFS and map-reduce

- Various sources, mostly external

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): HDFS + MapReduce

- 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
- MapReduce: A programming framework for processing parallelizable problems across huge datasets using a large number of commodity machines.

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

Colorado State University

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

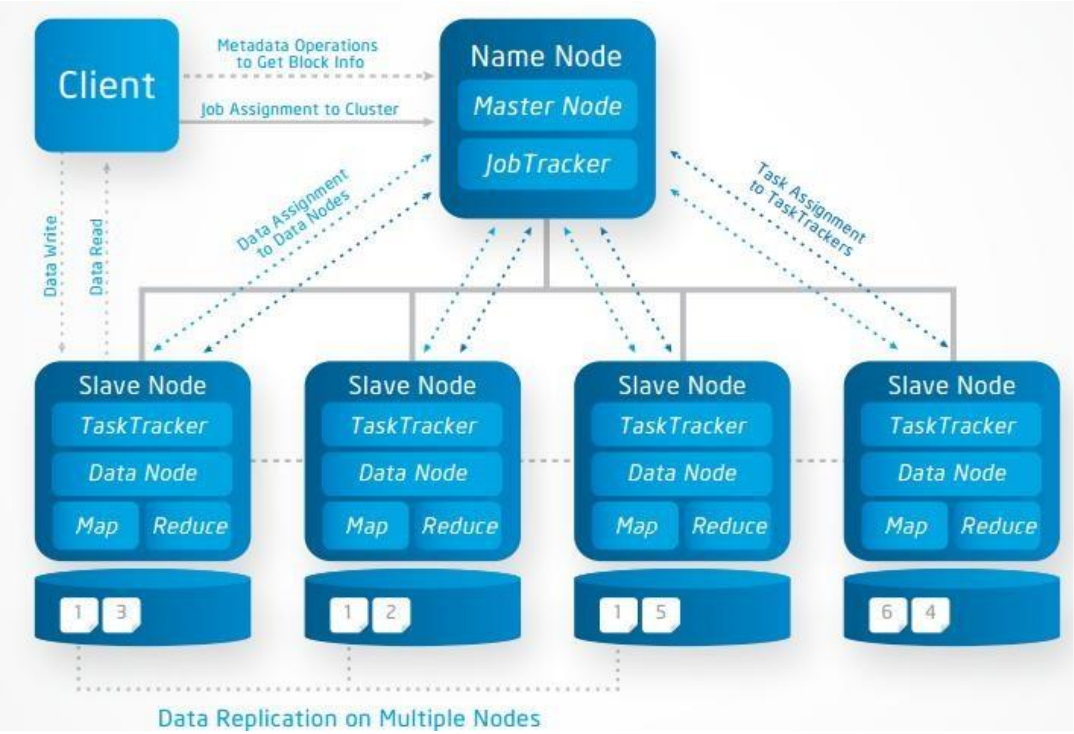
Metadata (corresponding to superblocks, Inodes)

- Name Node: metadata giving where blocks are physically located

Data (files blocks)

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

HDFS Architecture



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

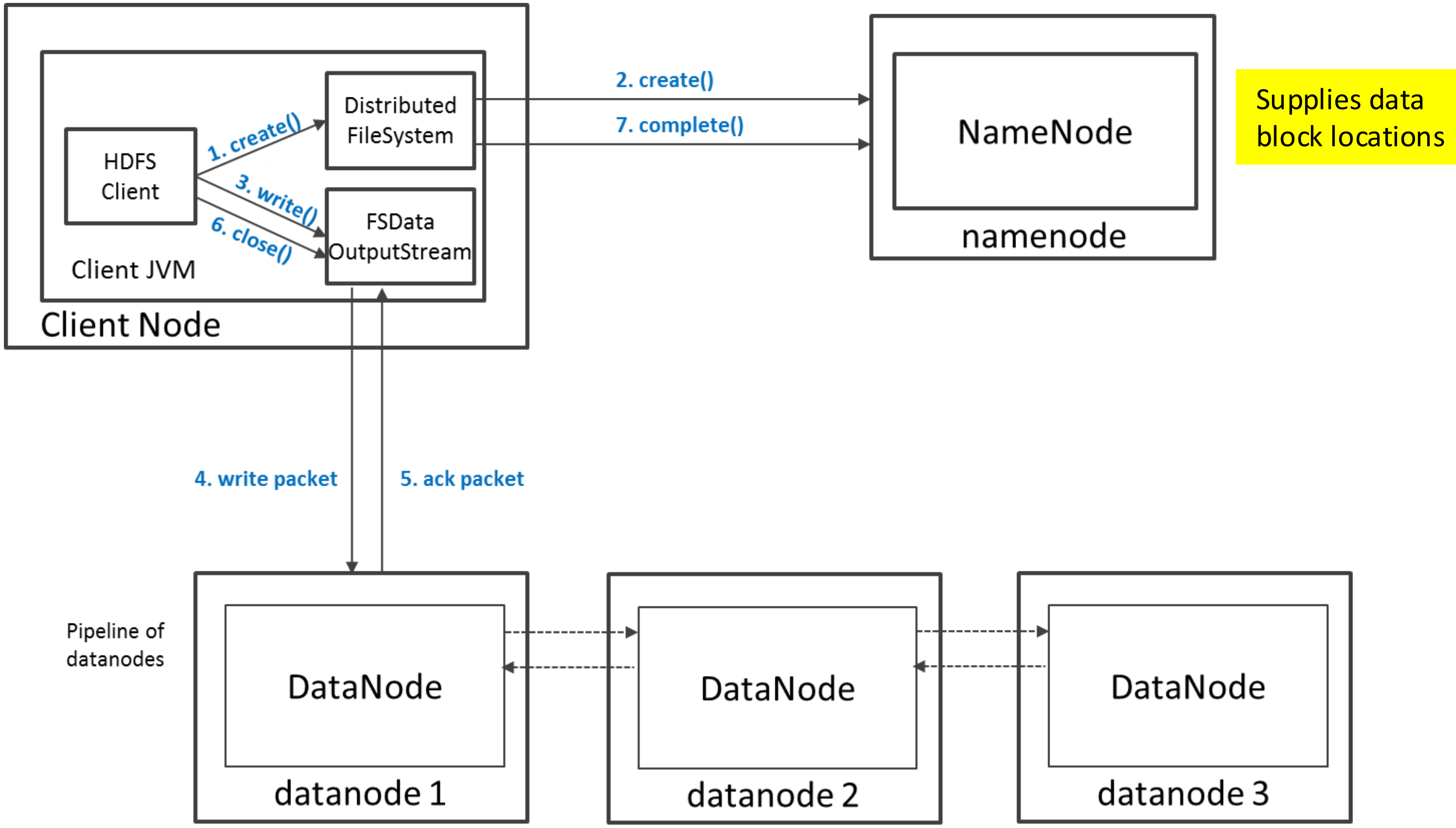
Secondary Name node
If primary fails.

Data is distributed and replicated.

Name Node: metadata giving where blocks are physically located
Data Nodes: hold blocks of files (files are distributed

Q. What do I need to know? motivation, approaches, concepts

HDFS Write operation



https://indico.cern.ch/event/404527/contributions/968835/attachments/1123385/1603232/Introduction_to_HDFS.pdf

CERN

HDFS Fault-tolerance

- Disks use error detecting codes to detect corruption.
- Individual node/rack may fail.
- Data Nodes (on slave nodes):
  - data is replicated. Default is 3 times. Keep a copy far away.
  - Send periodic heartbeat (I'm OK) to Name Nodes. Perhaps once every 10 minutes.
  - Name node creates another copy if no heartbeat.

HDFS Fault-tolerance

Name Node (on master node) Protection:

- Transaction log for file deletes/adds, etc. Creation of more replica blocks, when necessary, after a Data Node failure
- Standby name node: namespace backup
  - In the event of a failover, the Standby will ensure that it has read all of the edits from the Journal Nodes and then promotes itself to the Active state
  - Implementation/delay version dependent

Name Node metadata is in RAM as well as checkpointed on disk.
On disk the state is stored in two files:
- fsimage: Snapshot of file system metadata
- editlog: Changes since last snapshot

HDFS Command line interface

- hadoop fs –help
- hadoop fs –ls : List a directory
- hadoop fs mkdir : makes a directory in HDFS
- hadoop fs –rm : Deletes a file in HDFS
- copyFromLocal : Copies data to HDFS from local filesystem
- copyToLocal : Copies data to local filesystem
- Java code can read or write HDFS files (URI) directly

HDFS is on top of a local file system

https://hadoop.apache.org/docs/r2.4.1/hadoop-project-dist/hadoop-common/FileSystemShell.html

Distributing Tasks

MapReduce Engine:

- JobTracker splits up the job into smaller tasks("Map") and sends it to the TaskTracker process in each node.
- TaskTracker reports back to the JobTracker node and reports on job progress, sends partial results ("Reduce") or requests new jobs.
- Tasks are run on local data, thus avoiding movement of bulk data.
- Originally developed for search engine implementation.

Hadoop Ecosystem Evolution

- Hadoop YARN: A framework for job scheduling and cluster resource management, can run on top of Windows Azure or Amazon S3.
- Apache spark is more general, faster and easier to program than MapReduce.
    - Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing, Berkeley, 2012