

CS 370: OPERATING SYSTEMS
[MASS STORAGE]

Shrideep Pallickara
 Computer Science
 Colorado State University

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.1
 Dept. Of Computer Science, Colorado State University

Frequently asked questions from the previous class survey

- How does NTFS compare with UFS?

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.2
 Professor: SHRIDEEP PALICKARA Dept. Of Computer Science, Colorado State University

Topics covered in this lecture

- Wrap-up of iNodes
- Flash Memory
- RAID
- Final Exam

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.3
 Professor: SHRIDEEP PALICKARA Dept. Of Computer Science, Colorado State University

inode: A quantitative look
 BLOCK Size = 8 KB and Pointers = 4 bytes

The diagram shows an inode structure with the following components and calculations:

- File Attributes:** 68 bytes
- Direct pointers to first few file blocks:** 128 - 68 = 12 = 48
- Number of direct pointers?** 48/4 = 12
- Single indirect pointer:** 3x4 = 12 bytes
- Double indirect pointer:** 3x4 = 12 bytes
- Triple indirect pointer:** 3x4 = 12 bytes

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.4
 Professor: SHRIDEEP PALICKARA Dept. Of Computer Science, Colorado State University

inode: A quantitative look
 BLOCK Size = 8 KB and Pointers = 4 bytes

- 12 **direct** pointers to file blocks
- Each file block = 8KB
- Size of file that can be represented with direct pointers
 - 12 x 8 KB = 96 KB

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.5
 Professor: SHRIDEEP PALICKARA Dept. Of Computer Science, Colorado State University

inode

The diagram illustrates an i-Node structure with 'Attributes' and pointers to different types of indirect blocks:

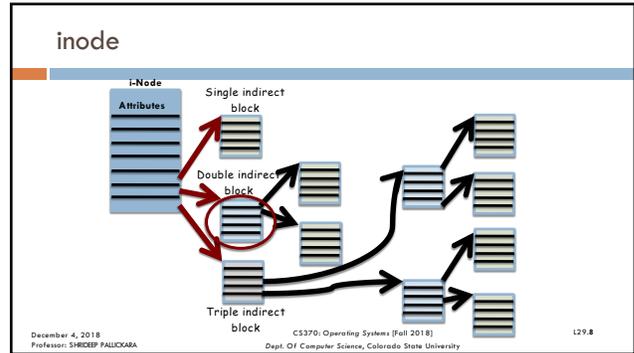
- Single indirect block:** A block containing pointers to file blocks.
- Double indirect block:** A block containing pointers to other indirect blocks.
- Triple indirect block:** A block containing pointers to other indirect blocks.

December 4, 2018 CS370: Operating Systems (Fall 2018) L29.6
 Professor: SHRIDEEP PALICKARA Dept. Of Computer Science, Colorado State University

inode: A quantitative look
 BLOCK Size = 8 KB and Pointers = 4 bytes

- Block size = 8 KB
- Single indirect block = block size = 8 KB (8192 bytes)
- Number of pointers held in a single-indirect-block
 - Block-size/Pointer-size
 - $8192/4 = 2048$
- With single-indirect pointer
 - Additional $2048 \times 8 \text{ KB} = 2^{11} \times 2^3 \times 2^{10} = 2^{24}$ (16 MB) of a file can be addressed

December 4, 2018
 Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University L29.7



inode: A quantitative look
 BLOCK Size = 8 KB and Pointers = 4 bytes

- With a **double indirect pointer** in the inode
 - The double-indirect block has 2048 pointers
 - Each pointer points to a different single-indirect-block
 - So, there are 2048 single-indirect blocks
 - Each single-indirect block has 2048 pointers to file blocks
- Double indirect addressing allows the file to have an additional size of
 - $2048 \times 2048 \times 8 \text{ KB} = 2^{11} \times 2^{11} \times 2^{13} = 2^{35}$ (32 GB)

December 4, 2018
 Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University L29.9

inode: A quantitative look
 BLOCK Size = 8 KB and Pointers = 4 bytes

- **Triple indirect addressing**
 - Triple indirect block points to 2048 double indirect blocks
 - Each double indirect block points to 2048 single indirect block
 - Each single direct block points to 2048 file blocks
- Allows the file to have an additional size of
 - $2048 \times 2048 \times 2048 \times 8 \text{ KB} = 2^{11} \times 2^{11} \times 2^{11} \times 2^{13} = 2^{46}$ (64 TB)

December 4, 2018
 Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University L29.10

Limits of triple indirect addressing

- In our example:
 - There can be $2048 \times 2048 \times 2048$ data blocks
 - i.e., $2^{11} \times 2^{11} \times 2^{11} = 2^{33}$
 - Pointers would need to be longer than 32-bits to fully address this storage

December 4, 2018
 Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University L29.11

What if we increase the size of the pointers to 64-bits (data block is still 8 KB) ?

- What is the maximum size of the file that we can hold?
- 8 KB data block can hold $(8192/8) = 1024$ pointers
- **Single indirect** can add
 - $1024 \times 8 \text{ KB} = 2^{10} \times 2^3 \times 2^{10} = 2^{23}$ (8MB) of additional bytes to the file

December 4, 2018
 Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University L29.12

What if we increase the size of the pointers to 64-bits (data block is still 8 KB)?

- **Double indirect** addressing allows the file to have an additional size of
 - $1024 \times 1024 \times 8 \text{ KB} = 2^{10} \times 2^{23} = 2^{33} \dots (8 \text{ GB})$
- **Triple indirect** addressing allows the file to have an additional size of
 - $1024 \times 1024 \times 1024 \times 8 \text{ KB} = 2^{10} \times 2^{33} = 2^{43} (8 \text{ TB})$

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.13

FLASH MEMORY

December 4, 2018

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.14

Flash memory is a type of a solid state storage

- No moving parts ... and stores data using electrical circuits
 - Can have better random I/O performance than HDDs, use less power, and is less vulnerable to physical damage
 - But significantly more expensive per byte

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.15

Transistors

- It takes one transistor to store a bit
- Ordinary transistors are electronic switches
 - Turned on and off by electricity
- Strength: Computer can store information simply by **passing patterns of electricity** through its memory circuits
- Weakness: As soon as power is turned off, transistors revert to their original state (loses all information)
 - Electronic amnesia!

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.16

Transistors in flash memory

The source and drain regions are rich in electrons (n-type silicon)
Electrons cannot flow from source to drain, because of the electron-deficient p-type material between them

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.17

A gate that floats?

- The extra gate in our transistor “floats” — it is not connected to any circuit
- Since the floating gate is entirely surrounded by an **insulator**, it will hold an electrical charge for months or years without requiring any power
- Even though the floating gate is not electrically connected to anything, it can be charged or discharged
 - Via **electron tunneling** by running a sufficiently high-voltage current near it

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.18

Transistors in flash memory

The diagram shows a cross-section of a flash transistor. A central floating gate is sandwiched between a control gate (top) and a bitline (right). A wordline is positioned above the control gate. The bitline is connected to an n-type region, and the control gate is connected to a wordline. The floating gate is also connected to an n-type region. The wordline is labeled with a '+' sign, and the bitline is labeled with a '+' sign. The n-type regions are labeled 'n'. The floating gate is labeled 'float gate'. The control gate is labeled 'control gate'. The wordline is labeled 'wordline'. The bitline is labeled 'bitline'. The ground is labeled 'ground'.

The presence of electrons on the floating gate is how a flash transistor stores a **one**.
Electrons stay there indefinitely, even when positive voltages are removed AND whether power is supplied to the unit or not.
Electrons can be flushed out by putting a negative voltage on the wordline. REPELS electrons back.

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.19

Flash storage: Erasure blocks

- Before flash memory can be written, it must be **erased** by setting each cell to a logical "1"
- Can only be erased in large units called **erasure blocks** (128-512 KB)
- Slow operation: takes several milliseconds
- Erasing an erasure block is what gives "flash memory" its name ...
 - Resemblance to the flash of a camera

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.20

Write page and read page

- Write Page:**
 - Once erased, flash memory can be written on a page-by-page basis
 - Each page is typically 2-4 KB
 - Writing a page takes about 10s of microseconds
- Read page**
 - Flash memory is read on a page-by-page basis
 - Reading a page takes about 10s of microseconds

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.21

Challenges in writing to a page

- To write a page, its **entire erasure block** must first be erased
 - Erasure is slow and affects a large number of pages
- Flash translation layer (FTL)
 - Maps logical flash pages to different physical pages on the flash device
 - When logical page is overwritten, the FTL writes the new version to a free, already-erased physical page
 - ... and remaps logical page to that physical page
 - Write remapping significantly improves performance

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.22

Durability [1/2]

- Normally, flash memory can retain state for months or years without power
- However, **high current loads** from flashing and writing memory **causes circuits to degrade**
 - After a few 1000~1,000,000 erase cycles a cell may wear out ... cannot reliably store a bit

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.23

Durability [2/2]

- Reading a flash memory cell a large number of times causes surrounding cells' charges to be **disturbed**
 - Read disturb error:** Location in memory read too many times without surrounding memory being rewritten

December 4, 2018
Professor: SHRIDEEP PALICKARA
CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University
L29.24

Improving durability

- Error correcting codes
- Bad page and bad erasure block management
 - Firmware stops storing data on defective blocks
- **Wear leveling**
 - Move logical pages to different physical pages to ensure *no physical page gets inordinate number of writes* and wears out prematurely
 - Some algorithms also migrate unmodified pages to protect against read disturb errors
- Spare pages and erasure blocks

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.25

Parameters for the Intel 710 Series SSD

- Capacity 300 GB
- Page Size 4 KB
- Performance
 - Bandwidth (Sequential Reads) 270 MB/s
 - Bandwidth (Sequential Writes) 210 MB/s
 - Read/ Write Latency 75 μ s
 - Random Reads Per Second 38,500
 - Random Writes Per Second 2,000
 - 2,400 with 20% space reserve

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.26

Parameter for the Intel 710 Series SSD

- Interface SATA 3 Gb/ s
- Endurance
 - Endurance 1.1 PB
 - 1.5 PB with 20% space reserve
- Power
 - Power Consumption Active/ Idle 3.7 W / 0.7 W

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.27

RAID STRUCTURE

December 4, 2018

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.28

RAID involves using large number of disks in parallel

- Improves **rate** at which data be can read/written
- Increases **reliability** of storage
 - Redundant information can be stored on multiple disks
 - Failure of 1 disk should not result in loss of data
- Redundant Array of **Independent** ~~Inexpensive~~ Disks

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.29

RAID levels

- Standardized by the Storage Networking Industry Association (SNIA)
 - In the Common RAID Disk Drive Format (DDF) standard
- Originally there were 5 levels
- There are other nested levels

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.30

Reliability through redundancy

- Store information that is not normally needed
- Can be used in the event of disk failure
 - **Rebuild** lost information
- Simplest approach: **Mirroring**
 - Duplicate every disk
 - Data lost only if 2nd disk fails BEFORE 1st one is replaced
 - Watch for: Correlated failures

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.31

RAID parallelism

- **Stripe** data across disks
- Objectives
 - ① Increase throughput
 - ② Reduce response times of large accesses

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.32

RAID Parallelism: Stripe data across disks Bit level striping

- **Split bits** of each byte across multiple disks
 - 8 disks: Bit **i** of each byte written to disk **i**
 - Bit 3 written to disk 3
- Array of 8 disks treated as a single disk
 - 8 times the access rate
 - Every disk participates in every read/write

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.33

RAID Parallelism: Block-level striping

- **Blocks** of a file are striped across multiple disks
- When there are **n** disks
 - Block **i** of the file written to ...
 - Disk: $(i \bmod n) + 1$
 - 4 disks: Block 9 of file goes to disk 2
 - 4 disks: Block 8 of file goes to disk 1

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.34

RAID levels

- Striping improves transfer rates
 - BUT not reliability
- Disk striping usually combined with **parity**
- Different schemes classified according to levels
 - RAID levels

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.35

RAID 0: Stripe blocks without redundancy



- No mirroring
- No parity

December 4, 2018
Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University

L29.36

RAID 1: Disk mirroring



- Each disk is mirrored

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.37

RAID 2: Memory style error correcting code

- Parity bit records number of 1 bits in byte
 - Even: parity 0
 - Odd: parity 1
- Use to detect single-bit errors
- Error correcting schemes
 - 2 or more extra bits to recover from single-bit errors

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.38

RAID 2: Error Correcting Codes



Error correction bits

- If one disk fails:
 - Remaining bits of the byte + error correction bits
 - Read from other disks
 - Reconstruct damaged data

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.39

RAID 3: Single parity bit used for error correction

- We can identify damaged sector
- Figure out if any bit in sector is 0 or 1
 - Compute parity of corresponding bits from other sectors
 - If parity of remaining bits == stored parity
 - Missing bit = 0
 - Otherwise, missing bit = 1

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.40

RAID 3: Single parity bit used for error correction



Error correction bits

Issues

- Fewer I/Os per-second since every disk participates in every I/O
- Overheads for computing parity bits

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.41

RAID-4 Block interleaved parity

- Block-level striping
 - Block read accesses only one disk
 - Data transfer rate slower for each access
 - Multiple reads proceed in parallel
 - Higher overall I/O rate
- Parity block on a separate disk

December 4, 2018
 Professor: SHRIDEEP PALICKARA

CS370: Operating Systems (Fall 2018)
 Dept. Of Computer Science, Colorado State University

L29.42

RAID 4: Block interleaved parity



Parity block

If one disk fails

- Parity block used with corresponding blocks
- Restore blocks of failed disk

December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.43

RAID-5 Block interleaved distributed parity

- Spread data and parity among all $N+1$ disks
- Avoid overuse of single parity disk
- Parity block does not store parity for blocks on the same disk

December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.44

RAID 5: Block interleaved distributed parity



December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.45

RAID-6

- Store extra redundant information
- Guard against **multiple** disk failures
- Error correcting codes are used
- Reed-Solomon codes
- 2-bits of redundant data
- For every 4-bits of data

December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.46

RAID-6



December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.47

In the computer science department

- RAID 1
 - To mirror the root disks of the servers
- RAID 5
 - For all the "no_back_up" partitions
- RAID 6
 - For all data disks

December 4, 2018
Professor: SHRIDEEP PALICKARA CS370: Operating Systems (Fall 2018)
Dept. Of Computer Science, Colorado State University L29.48

FINAL EXAM

December 4, 2018

CS370: Operating Systems [Fall 2018]
Dept. Of Computer Science, Colorado State University

L29.49

Final exam

- In our regular classroom (Clark A-103) from
 - There will be NO MAKEUP exam
- Comprehensive exam
 - Covers **ALL** topics that we have discussed in the course
- Duration: 2 hours

December 4, 2018

Professor: SHRIDEEP PALICKARA

CS370: Operating Systems [Fall 2018]
Dept. Of Computer Science, Colorado State University

L29.50

Breakdown

□ Processes and IPC	5
□ Threads:	10
□ CPU Scheduling	10
□ Process Synchronization & Atomic Transactions	15
□ Deadlocks	10
□ Memory Management	10
□ Virtual Memory	10
□ Virtualization	10
□ File Systems	15
□ Mass Storage & Disk Scheduling	5

December 4, 2018

Professor: SHRIDEEP PALICKARA

CS370: Operating Systems [Fall 2018]
Dept. Of Computer Science, Colorado State University

L29.51

The contents of this slide-set are based on the following references

- *Avi Silberschatz, Peter Galvin, Greg Gagne. Operating Systems Concepts, 9th edition. John Wiley & Sons, Inc. ISBN-13: 978-1118063330. [Chapter 10, 11]*
- *Andrew S Tanenbaum and Herbert Bos. Modern Operating Systems. 4th Edition, 2014. Prentice Hall. ISBN: 013359162X/ 978-0133591620. [Chapter 5]*
- *Thomas Anderson and Michael Dahlin. Operating Systems: Principles & Practice. 2nd edition. ISBN: 978-0-9856735-2-9 [Chapter 12]*
- *Chris Woodford. Flash Memory. <http://www.explainthatstuff.com/flashmemory.html>*

December 4, 2018

Professor: SHRIDEEP PALICKARA

CS370: Operating Systems [Fall 2018]
Dept. Of Computer Science, Colorado State University

L29.52