Operating Systems

Tutorial 2 & 16

Michael Tänzer

os-tut@nhng.de
http://os-tut.nhng.de

Calendar Week 5

Outline

True or False

- When using linked allocation files can only be accessed sequentially
- When using inodes it doesn't matter whether blocks are allocated contiguously or not
- The file size is stored in the inode

True or False

- When using linked allocation files can only be accessed sequentially
- When using inodes it doesn't matter whether blocks are allocated contiguously or not
- The file size is stored in the inode

True or False

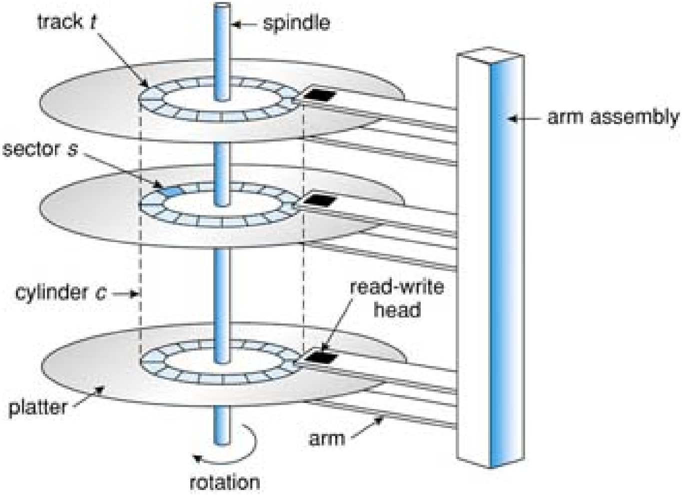
- When using linked allocation files can only be accessed sequentially
- When using inodes it doesn't matter whether blocks are allocated contiguously or not
- The file size is stored in the inode

True or False

- When using linked allocation files can only be accessed sequentially
- When using inodes it doesn't matter whether blocks are allocated contiguously or not
- The file size is stored in the inode

Explain the terms cylinder, track and sector

Explain the terms cylinder, track and sector



Estimate the sustained transfer rate

Ignore the time to move to the next track and assume no initial seek is required

Hard Disk

- 7200 RPM
- 512 bytes sector size
- 160 sectors per track

Estimate the sustained transfer rate

Ignore the time to move to the next track and assume no initial seek is required

Hard Disk

- 7200 RPM
- 512 bytes sector size
- 160 sectors per track

$$7200 \text{ RPM} = \frac{7200}{60} \text{ rounds/s} = 120 \text{ tracks/s}$$

$$1 \text{ track} = 160 \text{ sectors} \cdot 512 \text{ bytes/sector} = 81920 \text{ bytes}$$

$$\Rightarrow \text{transfer rate} = 120 \text{ tracks/s} \cdot 81920 \text{ bytes/track} = 9600 \text{ KB/s}$$

Explain the term ‘sector sparing’

What problem can occur when it’s used and how can it be mitigated?

Explain the term ‘sector sparing’

What problem can occur when it’s used and how can it be mitigated?

- Disk contains spare sectors which are hidden to the OS
- When a defective sector is detected the disk controller replaces the bad sector with a spare

⇒ Future requests to the bad sector are redirected to the spare

- The mapping is transparent to the OS

Explain the term ‘sector sparing’

What problem can occur when it’s used and how can it be mitigated?

- Disk contains spare sectors which are hidden to the OS
- When a defective sector is detected the disk controller replaces the bad sector with a spare

⇒ Future requests to the bad sector are redirected to the spare

- The mapping is transparent to the OS

– Real structure differs from the structure the OS ‘sees’

⇒ The disk scheduler of the OS could make a decision which would be good in theory but is far from optimal in reality

- One could have some spare sectors on each track so the difference doesn’t become very big

Explain the term sector slipping

Explain the term sector slipping

- Similar to sector sparing
- Instead of only remapping the bad sector to the spare one all sectors behind the bad sector are remapped one spot (until the cascade reaches a spare sector)

⇒ The bad sector is mapped to the sector directly behind it

\+ The difference between the abstract disk layout and the real one is only one sector offset

Compute the average track head movements using FCFS/FIFO, Scan and SSTF

- Initial head position: track 100 moving towards track 0
- Track requests: 129, 37, 31, 99, 89, 102, 15, 63, 130

Compute the average track head movements using FCFS/FIFO, Scan and SSTF

- Initial head position: track 100 moving towards track 0
- Track requests: 129, 37, 31, 99, 89, 102, 15, 63, 130

FCFS		Scan		SSTF	
request	delta	request	delta	request	delta
100		100		100	
129	29	99	1	99	1
37	92	89	10	102	3
31	6	63	26	89	13
99	68	37	26	63	26
89	10	31	6	37	26
102	13	15	16	31	6
15	87	102	87	15	16
63	48	129	27	129	114
130	67	130	1	130	1
avg.:	46.67		22.22		22.89

Swap space in file vs. separate partition

Swap space in file vs. separate partition

Swap File

- \+ Can be accessed like a normal file ⇒ easier to implement
- \+ Can grow and shrink on demand
- – Each access is subject to the normal file operations ⇒ more overhead
- – Might get fragmented (especially if size is dynamic)

Swap Partition

- \+ Raw block access possible ⇒ less overhead
- \+ Data placement in the partition can be optimized for speed (no safety needed)
- – Fixed size

What's anonymous memory?

Why can non-anonymous memory be handled differently with respect to swapping?

What's anonymous memory?

Why can non-anonymous memory be handled differently with respect to swapping?

- Anonymous memory are those memory regions which weren't directly loaded from a file on the file system (i. e. stack, heap and uninitialised data)
- Non-anonymous memory is associated with a file (e. g. the application's binary, a library, a memory mapped file)

⇒ If a non-anonymous page is chosen for eviction it doesn't need to be swapped out to the global swap area but the associated file can serve as swap area

- Exception: modified code (binaries and libraries) should not (and probably can't due to missing privileges) be written back to the original file but to the global swap area

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

Criteria

a) How many disks do you need?

b) You want to modify one byte of data. How many blocks do you have to read/write?

c) One of the data disks fails. What has to be done to recover the data?

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

Abbrevations

SLED	Single Large Expensive Disk
RAID	Redundant Array of Inexpensive Disks
LBN	Logical Block Number
(d, PBN)	Physical Block Number on disk d

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

SLED

a) 1 Disk
b) 1 read, 1 write
c) Recovery not possible

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 0

- Block-striping: each block is mapped to one of n disks, e. g. $(d, \text{PBN}) := (\text{LBN} \mod n, \text{LBN} \div n)$
- Blocks can be accessed in parallel $\Rightarrow$ high throughput on read and write
- Total size $= \sum$ disk sizes

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 0

- Block-striping: each block is mapped to one of n disks, e. g. $(d, \text{PBN}) := (\text{LBN} \mod n, \text{LBN} \div n)$
- Blocks can be accessed in parallel $\Rightarrow$ high throughput on read and write
- Total size $= \sum$ disk sizes

a) 4 disks

b) 1 read, 1 write

c) Recovery not possible

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 1

- Mirroring, data is written to *n* disks
- Only 1 disk needed to read $\Rightarrow$ reads can be performed in parallel $\Rightarrow$ high throughput on read
- Total size $=$ min(disk sizes)

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 1

- Mirroring, data is written to *n* disks
- Only 1 disk needed to read ⇒ reads can be performed in parallel ⇒ high throughput on read
- Total size = min(disk sizes)

a) 8 disks

b) 1 read, 2 writes (data + mirror)

c) Read data from mirror disk

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 2

- Bit-striping + ECC
- $n - \log_2(n)$ disks for data and $\log_2(n)$ disks for ECC

$\Rightarrow$ longer data words/more disks give higher data/ECC ratio

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 2

- Bit-striping + ECC
- $n - \log_2(n)$ disks for data and $\log_2(n)$ disks for ECC

$\Rightarrow$ longer data words/more disks give higher data/ECC ratio

a) 7 disks

b) 4 reads, 7 writes (bits of a word are spread over all disks)

c) Reconstruct data using hamming code

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 3

- Bit-striping + parity
- Less secure than RAID 2
- Total size $= (n - 1)$ disk size

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 3

- Bit-striping + parity
- Less secure than RAID 2
- Total size $= (n - 1)$ disk size

a) 5 disks

b) 4 read, 5 write (data + parity)

c) Reconstruct data using the parity disk

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 4

- Block-striping + parity
- Same security as RAID 3 but more efficient on read/write
- Only 1 disk needed to read
- Parity disk is accessed on every write $\Rightarrow$ bottleneck and may wear out fast
- Total size $= (n - 1)$ disk size

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 4

- Block-striping + parity
- Same security as RAID 3 but more efficient on read/write
- Only 1 disk needed to read
- Parity disk is accessed on every write $\Rightarrow$ bottleneck and may wear out fast
- Total size $= (n - 1)$ disk size

a) 5 disks

b) 2 read, 2 write (data + old/new parity)

c) XOR blocks on remaining disks

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 5

- Block-striping + distributed parity
- Like RAID 4 but parity blocks are distributed among all disks
- Load is balanced among the disks
- Total size $= (n - 1)$ disk size

Compare SLED and RAID 0 to 5

Each RAID uses 4 disks for actual storage and RAID 2 three bits for error correction

RAID 5

- Block-striping + distributed parity
- Like RAID 4 but parity blocks are distributed among all disks
- Load is balanced among the disks
- Total size $= (n - 1)$ disk size

a) 5 disks

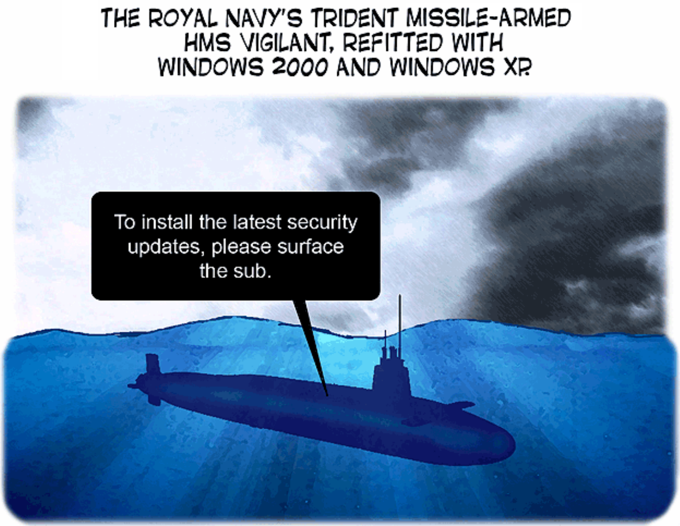
b) 2 read, 2 write (data + old/new parity)

c) XOR blocks on remaining disks

The Kernel is executing `f()` when an interrupt occurs. Can the interrupt handler safely call `f()` in any case?

Why do most modern OSs split handling of interrupts into two phases, a high- and a low-priority phase?

The End


THE ROYAL NAVY'S TRIDENT MISSILE-ARMED HMS VIGILANT, REFITTED WITH WINDOWS 2000 AND WINDOWS XP
To install the latest security updates, please surface the sub.