17+0.5 Bonus/20

Betriebssysteme WS 22/23

Blatt 09

Daniel Augustin, Malte Pullich

22.12.2022

4/5

1

a)

Da die Nummerierung mit 0 anfängt ist es allerds der Datenblock mit der 13 Nummer 13-1=12

Innerhalb des $[\frac{50000}{4096}] = 12$ten referenzierten Datenblocks im Speicher befindet sich das Byte Nummer 50000. Der Zeiger zu diesem Datenblock befindet sich damit innerhalb der von dem einfach indirekten Zeiger referenzierten Zeigern. welcher genau?
Ein Block mit indirekten Zeigern hat $\frac{4096}{4} = 1024$ Zeiger.

Ablauf:

- Das Betriebssystem berechnet, wo der Zeiger zu diesem Byte liegt. Hierbei wird der Zeiger 12 zu Block 12 gebraucht.
  Nr.
- Im I-Node liegen die Zeiger 0-9, dadurch und durch die Anzahl der Zeiger in einem Datenblock weiß das Betriebssystem, dass der Zeiger 12 im einfach indirekten Verweis liegt.
  ihr meint a) ist die Startadresse des ersten indirekten Zeigers
- Sei a die im einfach indirekten Zeiger liegende Adresse. Da es um den 13. Zeiger geht, berechnet das Betriebssystem die Adresse durch:
  a + 10 - 13 = a + 3
  2
  mit der Formel erhaltet ihr doch eine neative Zahl
  unpräzisse Ausdrucksweise
  -0.5 macht keinen Sinn
  12-10
- Das Betriebssystem ruft jetzt die Adresse des Blocks an dieser Adresse auf. Sei b nun die Adresse des Blocks. Dessen Daten beginnen bei Byte 49.152. Wir wollen allerdings Byte 50.000 abrufen. Das Betriebssystem berechnet diese Adresse durch:
  b + 50.000 - 49.152 = b + 848
- Das Betriebssystem kann nun diese Adresse zum Abruf des Bytes 50.000 verwenden.

b)

Im Dateisystem FAT32 werden Dateien als verkettete Liste gespeichert. Das Betriebssystem muss beim nullten Datenblock der Datei anfangen, um auf ein beliebiges Byte wahlfrei zugreifen zu können. Es folgt $N = [\frac{n}{b}]$ Verweisen der verketteten Liste. Wenn das Dateisystem bei diesem Block angelangt ist, berechnet es den Offset in den Block zu Byte n. Die Daten des Blocks selbst beginnen bei bei Byte N · b. Wir wollen allerdings auf Byte n zugreifen. Sei c die Adresse des Blocks von Byte n. Das Betriebssystem kann jetzt Byte n an der Adresse (c+n - N · b) abrufen. Durch das Vorwissen zu verketteten Listen aus Algorithmen und Datenstrukturen wissen wir, dass sich die Laufzeit auf ein beliebiges Element linear zur Anzahl der darin enthaltenen Elemente verhält. Damit liegt die Laufzeit unseres Dateisystems in O(N) und damit ferner auch in $O([\frac{n}{b}])$. Durch die Eigenschaften der Komplexität liegt sie damit auch in O(n).

wenn a die Startadresse des Zeigerblocks ist auf den der 1-fach indirekte Zeiger zeigt, dann muss man nur 2 weiter Jumpen um bei 3ten Zeiger zu sein bzw. derm Zeiger Nr. 2 (wenn man von 0 anfängt zu zählen)

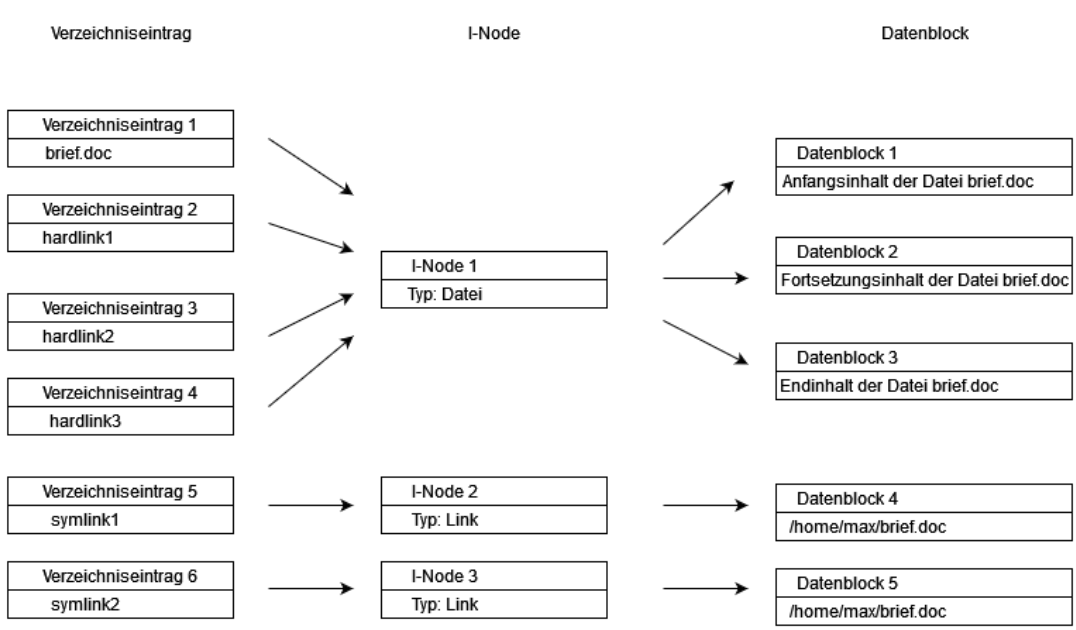
-1

-0.5 man muss -1 rechnen, da man zwischen n Objekten nur n-1 mal springen muss

6/8

2

2.1



wunderbar übersichtlich ^_^

2.2

+0.5 Bonuspunkte

	I-Node	Datenblock	Verzeichniseintrag
Datei	1	3	1
Symlinks	1	1	1
Hardlinks	0	0	1

=> 5 KiB Speicherplatz, 3 I-Nodes, 6 Verzeichniseinträge

2.3

Die Rechte der Datei brief.doc und die Hardlinks ändern sich simultan, da die Hardlinks auf den I-Node von Datei zugreifen.

Die Rechte der Systemlinks jedoch bleiben gleich, da diese eigene I-Nodes besitzen.

-2 2.4 fehlt

7/7

3

3.1

ps -aux

3.2

kill -<signal> <process id>

3.3

SIGSTOP:
Der Prozess wird pausiert.

SIGCONT
Kann einen Prozess wieder starten, mit SIGSTOP angehalten wurde, .

SIGTERM
Das Programm wird abgeschlossen und beendet. Laufende Berechnungen können hierbei noch fertig gestellt werden.

SIGKILL
Beendet den laufenden Prozess direkt.

perfekt

SIGKILL beendet den Prozess direkt, ohne notwendige Abläufe durchzuführen. Bei SIGTERM werden diese noch abgeschlossen

3.4

· screen -s countershell
· ./counter.sh
· Detach with "CTRL + A -> D"
· logout
· screen -r countershell
· exit