Hausaufgabeblatt 6

Gruppe 40:
Michelle Döring, Jannik Heinze, Iryna Novytska, Charlotte Fritsch

Aufgabe 6.5: Seitentauschverfahren

a) **Warum ist es sinnvoll, virtuelle Speicheradressen zu verwenden? Welche Aufgaben übernehmen die Seitentauschverfahren in diesem Zusammenhang?**

Mit virtuellen Adressen kann mehr Speicher angesprochen als physisch auf dem System installiert. Daten können vorübergehend aus dem RAM auf einen Disk-Speicher ausgelagert werden und somit der RAM künstlich vergrößert werden. Damit können Programme größer sein als der physisch vorhandene RAM oder mehrere Programme gleichzeitig arbeiten. Auch werden nicht immer alle Features deines Programms gleichermaßen genutzt und weniger genutzte Optionen können ausgelagert werden.
Die virtuelle Adresse muss natürlich in die physische Adresse übersetzt werden. Dabei werden Blöcke von in den meisten Fällen 4kB im virtuellen („Seite“) und physischen („Kachel“) Speicher einander zugeordnet. In der Seitentabelle steht neben anderen Informationen, wie diese einander zuzuordnen sind. Die Seiten werden dann nach Bedarf geladen (Demand Paging) oder falls das Prozessverhalten im Voraus bekannt ist so bereit gestellt, dass sie zum benötigten Zeitpunkt bereit stehen (Pre-Paging).
Quellen: https://www.computerweekly.com/de/definition/Virtueller-Arbeitsspeicher (abgerufen am 15.07.2020);
https://www.tutorialspoint.com/operating_system/os_virtual_memory.htm (abgerufen am 15.07.2020)

Sinvoll sind die virtuellen Adressen also deshalb, weil dadurch die Nutzung des physikalischen Speichers viel effizienter abläuft. Es werden nur Seiten/ Segmente geladen, die tatsächlich gebraucht werden. Dies fürt zu geringerem Speicherverbrauch.Es können mehrere Prozesse ausgeführt werden und deren Erzeugung läuft effizienter ab. Außerdem führt die Verwendung der virtuellen Adressen dazu, dass Prozesse gemeinsame Programmteile nutzen können.
(Quelle: http://www.inf.fu-berlin.de/lehre/WS11/OS/slides/OS_V15_Virtuelle_Speicher.pdf, abgerufen am: 16.07.2020)
Die Seitentauschverfahren sind dafür da, um Seitenfehler durch intelligentes Verdrängern zu minimieren, die während der Übersetzung der Prozess-Adressen in

die Hauptspeicheradressen entstehen können. (Quelle: Vorlesung 6, S.45)

b) **Gegeben ist ein physikalischer Speicher mit 4 Kacheln. Für folgende Seitenreihenfolge R=1,2,3,2,4,5,3,6,1,4,2,3,1,4 sind die jeweilige Speicher-Kachel-Zuordnung zu implementieren.**

Optimal:

Zeit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Zugriff Seite	1	2	3	2	4	5	3	6	1	4	2	3	1	4
Kachel 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Kachel 2		2	2	2	2	5	5	6	6	6	2	2	2	2
Kachel 3			3	3	3	3	3	3	3	3	3	3	3	3
Kachel 4						4	4	4	4	4	4	4	4	4

Optimal hat 3 Seitenfehler (ohne Initialisierungsfehler).

FIFO:

Zeit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Zugriff Seite	1	2	3	2	4	5	3	6	1	4	2	3	1	4
Kachel 1	1	1	1	1	1	5	5	5	5	5	5	3	3	3
Kachel 2		2	2	2	2	2	2	6	6	6	6	6	6	4
Kachel 3			3	3	3	3	3	3	1	1	1	1	1	1
Kachel 4					4	4	4	4	4	4	2	2	2	2
nächste Kachel	-	-	-	-	1	2	2	3	4	4	1	2	2	3

FIFO hat 6 Seitenfehler (ohne Initialisierungsfehler).

LFU
Bei gleicher Frequenz wird in Anlehnung an die Vorlesung (VL.6, S.49) als zweite Strategie FIFO ausgewählt.

Zeit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Zugr. Seite	1	2	3	2	4	5	3	6	1	4	2	3	1	4
Kachel 1	1	1	1	1	1	5	5	5	1	1	1	1	1	1
Kachel 2		2	2	2	2	2	2	2	2	2	2	2	2	2
Kachel 3			3	3	3	3	3	3	3	3	3	3	3	3
Kachel 4					4	4	4	6	6	4	4	4	4	4
Zähler														
1	1	1	1	1	1	(1)	(1)	(1)	2	2	2	2	3	3
2	-	1	1	2	2	2	2	2	2	2	3	3	3	3
3	-	-	1	1	1	1	2	2	2	2	2	3	3	3
4	-	-	-	-	1	1	1	(1)	(1)	2	2	2	2	3
5	-	-	-	-	-	1	1	1	(1)	(1)	(1)	(1)	(1)	(1)
6	-	-	-	-	-	-	-	1	1	(1)	(1)	(1)	(1)	(1)

LFU hat 4 Seitenfehler (ohne Initialisierungsfehler).

LRU:

Zeit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Zugriff Seite	1	2	3	2	4	5	3	6	1	4	2	3	1	4
Kachel 1	1	1	1	1	1	5	5	5	5	4	4	4	4	4
Kachel 2		2	2	2	2	2	2	6	6	6	6	3	3	3
Kachel 3			3	3	3	3	3	3	3	3	2	2	2	2
Kachel 4					4	4	4	4	1	1	1	1	1	1
Stapel 1	1	2	3	3	4	5	3	6	1	4	2	3	1	4
Stapel 2	-	1	2	2	3	4	5	3	6	1	4	2	3	1
Stapel 3	-	-	1	1	2	3	4	5	3	6	1	4	2	3
Stapel 4	-	-	-	-	1	2	2	4	5	3	6	1	4	2

LRU hat 6 Seitenfehler (ohne Initialisierungsfehler).

Aufgabe 6.6: Buddy Verfahren

In dieser Aufgabe wird die Belegung eines leeren 256 MB Speicherblocks simuliert. Die Anfragen und die Aufteilung nach dem Buddy Verfahren sind der Abbildung 1 zu entnehmen.

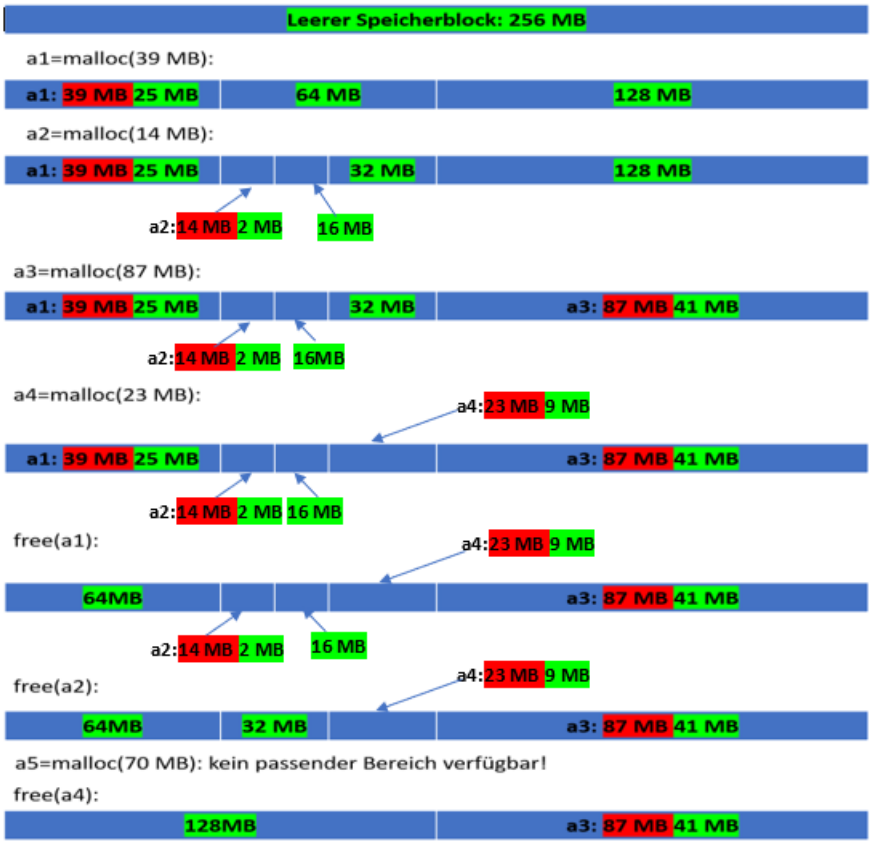

Abbildung 1: Handsimulation Buddy Verfahren

Aufgabe 6.7: FAT erstellen

Die gegebene Speicherbelegung entspricht folgender FAT:

Block	nächster Block
0	NULL
1	27
2	11
3	NULL
4	19
start A: 5	24
6	NULL
start E: 7	1
8	29
start C: 9	26
10	4
11	NULL
12	NULL
13	NULL
14	NULL
15	28
16	22
17	8
18	6
19	32
20	NULL
21	NULL
22	NULL
start D: 23	15
24	18
start B: 25	17
26	2
27	10
28	20
29	31
start F: 30	16
31	NULL
32	12