Stefan Fischer Benjamin Merle Kammer	1	2	3	Σ

Übungsblatt Nr. 2

(Abgabetermin 04.05.2017)

Aufgabe 1

a)

b)

c)

d)

e)

Aufgabe 2

Aufgabe 3

Laufzeit von Algorithmus A: $T_A(n) = (\log_8 n + 1) \cdot n$ mit $T_A(1) = 1$
Laufzeit von Algorithmus B: $T_B(n) = 8T_B(\frac{n}{8}) + n^\alpha$ mit $\alpha \in \mathbb{R}_+$

a)

Annahmen: n ist eine Achterpotenz und $T_B(1) = 1$

$$\begin{aligned}
T_B(n) &= 8T_B(\frac{n}{8}) + n^\alpha \\
&= 8(8T_B(\frac{n}{64}) + (\frac{n}{8})^\alpha) + n^\alpha \\
&= 8(8(8T_B(\frac{n}{512}) + (\frac{n}{64})^\alpha) + (\frac{n}{8})^\alpha) + n^\alpha \\
&\vdots \\
&= 8^i \cdot T_B(\frac{n}{8^i}) + \sum_{k=0}^{i-1}(8^k \cdot (\frac{n}{8^k})^\alpha) \\
&\text{für } i = \log_8 n \text{ gilt:} \\
&= 8^{\log_8 n} \cdot T_B(\frac{n}{8^{\log_8 n}}) + \sum_{k=0}^{(\log_8 n)-1}(8^k \cdot (\frac{n}{8^k})^\alpha) \\
&= n \cdot T_B(1) + \sum_{k=0}^{(\log_8 n)-1}(8^k \cdot (\frac{n}{8^k})^\alpha) \\
&\text{Da: } n = 8^x \text{ angenommen werden soll, gilt:} \\
&= 8^x + \sum_{k=0}^{(\log_8 8^x)-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha) \\
&= 8^x + \sum_{k=0}^{x-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha)
\end{aligned}$$

Behauptung: $T_B(n) = 8^x + \sum_{k=0}^{x-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha)$
Beweis durch vollständige Induktion:
Induktionsanfang: $T(1) = 1$, denn $8^0 + \sum_{k=0}^{0-1}(8^k \cdot (\frac{8^0}{8^k})^\alpha) = 1 + \sum_{k=0}^{-1}(8^k \cdot (\frac{1}{8^k})^\alpha) = 1$ ✓
Induktionsvorraussetzung: Sei $n = 8^x$ mit $x \in \mathbb{N}_0$ und es gelte $T(8^x) = 8^x + \sum_{k=0}^{x-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha)$
Induktionsbehauptung: Es gilt $T(8^{x+1}) = 8^{x+1} + \sum_{k=0}^{(x+1)-1}(8^k \cdot (\frac{8^x+1}{8^k})^\alpha)$

Beweis:

$$\begin{aligned}
T(8^{x+1}) &= 8T_B(\frac{8^{x+1}}{8}) + (8^{x+1})^\alpha \\
&= 8T_B(8^x\frac{8^1}{8}) + (8^{x+1})^\alpha \\
&= 8T_B(8^x) + (8^{x+1})^\alpha \\
&= 8(8^x + \sum_{k=0}^{x-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha)) + (8^{x+1})^\alpha \\
&= 8 \cdot 8^x + 8 \cdot \sum_{k=0}^{x-1}(8^k \cdot (\frac{8^x}{8^k})^\alpha) + (8^{x+1})^\alpha \\
&= 8^{x+1} + \sum_{k=0}^{x-1}(8^{k+1} \cdot (\frac{8^x}{8^k})^\alpha) + (8^{x+1})^\alpha \\
&= 8^{x+1} + \sum_{k=1}^{x}(8^{k+1-1} \cdot (\frac{8^x}{8^{k-1}})^\alpha) + (8^{x+1})^\alpha \\
&= 8^{x+1} + \sum_{k=1}^{x}(8^k \cdot (\frac{8^x \cdot 8^1}{8^k})^\alpha) + (8^{x+1})^\alpha \\
&= 8^{x+1} + \sum_{k=1}^{x}(8^k \cdot (\frac{8^{x+1}}{8^k})^\alpha) + (8^{x+1})^\alpha
\end{aligned}$$

Es gilt: $(8^{x+1})^\alpha = (8^0 \cdot (\frac{8^{x+1}}{8^0})^\alpha)$

Daher ziehen wir $(8^{x+1})^\alpha$ in die Summe und verringern die Startvariabel um Eins:

$$= 8^{x+1} + \sum_{k=0}^{x}(8^k \cdot (\frac{8^{x+1}}{8^k})^\alpha) \qquad \square$$

Somit ist bewiesen, dass $T_B(n) = 8T_B(\frac{n}{8}) + n^\alpha = n + \sum_{k=0}^{(\log_8 n)-1}(8^k \cdot (\frac{n}{8^k})^\alpha)$ mit $\alpha \in \mathbb{R}_+$ gilt.

b)

$T_A(n) = (\log_8 n + 1) \cdot n$ mit $T_A(1) = 1$ und $T_B(n) = n + \sum_{k=0}^{(\log_8 n)-1}(8^k \cdot (\frac{n}{8^k})^\alpha)$
Algorithmus A ist schneller als B $\forall \alpha > 1$.
Beweis:
Fall 1: $\alpha = 1$

$$
\begin{aligned}
T_B(n) &= n + \sum_{k=0}^{(\log_8 n)-1} (8^k \cdot (\frac{n}{8^k})^1) \\
&= n + \sum_{k=0}^{(\log_8 n)-1} (8^k \cdot \frac{n^1}{8^k \cdot 1}) \\
&= n + \sum_{k=0}^{(\log_8 n)-1} n \\
&= n + ((\log_8 n - 1) - 0 + 1)n \\
&= n + n \cdot \log_8 n = T_A(n)
\end{aligned}
$$

Für $\alpha = 1$ sind die Algorithmen A und B somit gleich schnell. Wir überprüfen daher im Folgenden die Fälle $\alpha < 1$ und $\alpha > 1$ um zu bestimmen, wann der Algorithmus A schneller als B ist.
Wir verwenden dabei das Mastertheorem um die Laufzeit von $T_B(n)$ in Abhängigkeit von α zu bestimmen: $T_B(n) = 8T_B(\frac{n}{8}) + n^\alpha \Rightarrow a = 8, b = 8, f(n) = n^\alpha$ und $\log_b a = \log_8 8 = 1$

Fall 2: $\alpha < 1$
$f(n^\alpha) = O(n^{1-\epsilon}) \overbrace{\Rightarrow}^{\text{MT Fall 1}} T(n) = O(n^1)$ d.h., dass $T_B(n) \in O(n)$
$T_A(n) = n \cdot \log_8 n + n \notin O(n) \Rightarrow T_A(n) > T_B(n)$ d.h., dass Algorithmus B für $\alpha < 1$ schneller läuft.

Fall 3: $\alpha > 1$
$f(n^\alpha) = O(n^{1+\epsilon})$ also Fall 3 des Mastertheorems. Wir überprüfen daher die folgende Bedingung:
$a \cdot f(\frac{n}{b}) \leq c \cdot f(n)$ für $c < 1$ und n genügend groß:
$8 \cdot f(\frac{n}{8}) \leq c \cdot f(n) \Leftrightarrow 8\frac{n^\alpha}{8^\alpha} \leq c \cdot n^\alpha \Leftrightarrow \frac{8^1 \cdot n^\alpha}{8^\alpha} \leq c \cdot n^\alpha \Leftrightarrow \frac{n^\alpha}{8^{\alpha-1}} \leq c \cdot n^\alpha$ für $\frac{1}{8^{\alpha-1}} < c < 1\surd$
$\overbrace{\Rightarrow}^{\text{MT Fall 3}} T(n) = O(n^\alpha)$ d.h. $T_B(n) \in O(n^\alpha)$ $T_A(n) = n \cdot \log_8 n + n \in O(n^\alpha)$
aber $n^\alpha \notin O(T_A(n)) \Rightarrow T_A(n) < T_B(n)$ d.h., dass Algorithmus A für $\alpha > 1$ schneller ist als Algorithmus B. $\square$