Alexandros Paraskevaidis
Elias Gestrich

1. Übungsblatt

Aufgabe 1 - Asymptotisches Wachstum

Die Reihenfolge ist: $b(n) \in o(d(n)), d(n) \in o(e(n)), e(n) \in o(a(n)), a(n) \in o(c(n)), c(n) \in o(f(n))$, da:

$$\begin{aligned}
\lim_{n\to\infty} \frac{b(n)}{d(n)} &= \lim_{n\to\infty} \frac{12 \cdot \frac{n}{\sqrt{n}}}{\sqrt{n} \cdot \log(42 \cdot n)} \\
&= \lim_{n\to\infty} \frac{12n}{n \cdot \log(42 \cdot n)} \\
&= \lim_{n\to\infty} \frac{12}{\log(42 \cdot n)} \\
&= \lim_{n\to\infty} 0,
\end{aligned}$$

$$\begin{aligned}
\lim_{n\to\infty} \frac{d(n)}{e(n)} &= \lim_{n\to\infty} \frac{\sqrt{n} \cdot \log(42 \cdot n)}{n - 213465 \cdot \log(n^5)} \\
&\overset{\text{l.H.}}{=} \lim_{n\to\infty} \frac{\frac{\log(42\cdot n)}{2\sqrt{n}} + \frac{42\sqrt{n}}{42n}}{1 - \frac{213465 \cdot 5n^4}{n^5}} \\
&= \lim_{n\to\infty} \frac{\log(42 \cdot n) + 2}{2\sqrt{n} - \frac{2134650}{\sqrt{n}}} \\
&= \lim_{n\to\infty} \frac{\sqrt{n} \cdot \log(42 \cdot n) + 2\sqrt{n}}{2 \cdot n - 2134650} \\
&\overset{\text{l.H.}}{=} \lim_{n\to\infty} \frac{\frac{\log(42\cdot n)}{2\sqrt{n}} + \frac{42\sqrt{n}}{42n} + \frac{1}{\sqrt{n}}}{2} \\
&= \lim_{n\to\infty} \frac{\log(42 \cdot n) + 4}{4\sqrt{n}} \\
&\overset{\text{l.H.}}{=} \lim_{n\to\infty} \frac{\frac{42}{42\cdot n}}{\frac{2}{\sqrt{n}}} \\
&= \lim_{n\to\infty} \frac{1}{2\sqrt{n}} \\
&= 0,
\end{aligned}$$

$$\begin{aligned}
\lim_{n\to\infty} \frac{e(n)}{a(n)} &= \lim_{n\to\infty} \frac{n - 213465 \cdot \log(n^5)}{16^{\log_4 n}} \\
&= \lim_{n\to\infty} \frac{n - 213465 \cdot 5 \cdot \log n}{4^{2\log_4 n}} \\
&= \lim_{n\to\infty} \frac{n - 213465 \cdot 5 \cdot \log n}{n^2} \\
&\overset{\text{l.H.}}{=} \lim_{n\to\infty} \frac{1 - \frac{213465\cdot 5}{n}}{2n} \\
&= \lim_{n\to\infty} \frac{n - 213465 \cdot 5}{2n^2} \\
&\overset{\text{l.H.}}{=} \lim_{n\to\infty} \frac{1}{4n} \\
&= 0,
\end{aligned}$$

$$\begin{aligned}\lim_{n\to\infty}\frac{a(n)}{c(n)} &= \lim_{n\to\infty}\frac{16^{\log_4 n}}{n^5+111}\\ &\overset{\text{l.H.}}{=} \lim_{n\to\infty}\frac{2n}{5n^4}\\ &= \lim_{n\to\infty}\frac{2}{5n^3}\\ &= 0,\end{aligned}$$

Für $c(n) \in o(f(n))$:

$$2^{\sqrt{n}} = \exp\left(\sqrt{n}\ln 2\right) = \sum_{k=0}^{\infty}\frac{(\ln 2)^k\sqrt{n}^k}{k!} \geq \frac{(\ln 2)^{12}\sqrt{n}^{12}}{12!} = \frac{(\ln 2)^{12}n^6}{12!}$$

Also wenn $c(n) \in o\left(\frac{(\ln 2)^{12}n^6}{12!}\right)$, dann gilt auch $c(n) \in o(f(n))$.

$$\begin{aligned}\lim_{n\to\infty}\frac{n^5-111}{\frac{(\ln 2)^{12}}{12!}n^6} &\overset{\text{l.H.}}{=} \lim_{n\to\infty}\frac{5n^4}{\frac{6(\ln 2)^{12}}{12!}n^5}\\ &= \lim_{n\to\infty}\frac{5\cdot 12!}{6(\ln 2)^{12}n}\\ &= 0\end{aligned}$$

■

Aufgabe 2 - Laufzeit

Algorithmus 1: $f_1(n) \in \mathcal{O}\left(\sqrt{n}\right)$,
Algorithmus 2: $f_2(n) \in = \left(\frac{n}{2}-1\right) = \mathcal{O}\left(\frac{n}{2}\right) = \mathcal{O}(n)$,
Algorithmus 3: $f_3(n) \in \mathcal{O}(n\cdot(n-1) = \mathcal{O}(n^2-n) = \mathcal{O}(n^2)$
Algorithmus 4: $f_4(n) \in \mathcal{O}(1)$ (für $\mathbb{N} = \{0,1,2,\dots\} : f_4(n) \in \mathcal{O}(2) = \mathcal{O}(1)$ Algorithmus 5: $f_5 \in \mathcal{O}(1)$, da für $n_1 := 1000001$ gilt, dass für $n \in \mathbb{N} : n \geq n_1 : f_5(n) = f_4(n) \in \mathcal{O}(1)$

Aufgabe 3 - Rekursion

a)
- **Vor.**: $T(0) = 0$ und $T(n) = 3\cdot T(n-1)+5$ für alle $n \geq 1$
  Beh.: Für $n \geq 1$ gilt $T(n) = 2.5\cdot(3^n-1)$
  Bew.:

  I.A. $n = 1 : T(n) = T(1) = 3\cdot T(0)+5 = 3\cdot 0+5 = 5 = 2.5\cdot 2 = 2.5\cdot(3-1) = 2.5\cdot(3^n-1)$

  I.S. $n \curvearrowright n+1$

  I.V. $T(n) = 2.5\cdot(3^n-1)$

  Zu zeigen $T(n+1) = 2.5\cdot(3^{n+1}-1)$:

$$\begin{aligned}T(n+1) &= 3\cdot T(n)+5 && \mid \text{ I.V.}\\ &= 3\cdot(2.5\cdot(3^n-1))+5\\ &= 2.5\cdot(3\cdot 3^n-3)+2.5\cdot 2\\ &= 2.5\cdot(3^{n+1}-1)\end{aligned}$$

■

- **Vor.**: $T(n) = 1$ für $n \leq 3$ und $T(n) = T(n-1)+\frac{n}{3}$ sonst
  Beh.: Für $n \geq 1$ gilt $T(n) \leq n^2$
  Bew.:

  I.A. $n = 1 : T(n) = T(1) = 1 \leq 1^2 = n^2$

  I.S. $n \curvearrowright n+1$

  I.V. $T(n) \leq n^2$

Zu zeigen $T(n+1) \leq (n+1)^2$:

$$\begin{aligned} T(n+1) &= T(n) + \frac{n}{3} && \mid \text{ I.V.} \\ &\leq n^2 + \underbrace{\frac{n}{3}}_{\leq 2n+1} \\ &\leq n^2 + 2n + 1 \\ &\leq (n+1)^2 \end{aligned}$$

$\blacksquare$

- **Vor.**: $T(1) = 1$ für $n = 1$ und $T(n) = T(\frac{n}{2}) + 2$
  Beh.: Für $n \in \{2^i : i \in \mathbb{N}_0\}$ gilt $T(n) \leq 2\log_2(n) + 1$
  Also für $i \in \mathbb{N}_0$ gilt $T(2^i) \leq 2\log_2(2^i) + 1 = 2 \cdot i \cdot \log_2(2) + 1 = 2i + 1$
  Bew.:

  I.A. $i = 0 : T(2^i) = T(2^0) = T(1) = 1 \leq 1 = 2 \cdot 0 + 1 = 2i + 1$

  I.S. $i \curvearrowright i + 1$

  I.V. $T(2^i) \leq 2i + 1$

  Zu zeigen $T(2^{i+1}) \leq 2(i+1) + 1$:

$$\begin{aligned} T(2^{i+1}) &= T(\frac{2^{i+1}}{2}) + 2 \\ &= T(2^i) + 2 && \mid \text{ I.V.} \\ &\leq 2i + 1 + 2 \\ &\leq 2(i+1) + 1 \end{aligned}$$

$\blacksquare$

b) $T(n) = 1$ für $n \leq 42$, $T(n) = T(n-1) + 1$ sonst.

Aufgabe 4 - Lahme Schafe

Man kann eine Art "Bubble-Sort" verwenden. Also man sagt das erste Schaf wäre das langsamste und vergleicht dieses dann mit dem nächstem Schaf, schaut welches langsamer ist, sagt dieses wäre das langsamste und vergleicht das langsamste Schaf wieder mit dem nächsten Schaf usw.

Algorithmus 1: *find_slowest_sheep*(*Sheep*[ ] *sheep*)

$slowest\_sheep \leftarrow sheep[0]$
for $i = 1; i < sheep.count;$ **do**
  $slowest\_sheep \leftarrow slower\_sheep(slowest\_sheep, sheep[i]$
return *slowest_sheep*

Für n Schafe ist die Laufzeit also $1 + n - 1 = n$, folglich linear. Das funktioniert, da uns nur das langsamste Schaf interessiert und nicht wie schnell jedes Schaf im Vergleich zu jedem anderem Schaf ist. Wenn wir so also wissen, dass Schaf 3 schneller ist als Schaf 4, dann ist es irrelevant zu wissen ob Schaf 5 langsamer ist als Schaf 3, da dies nichts darüber aussagt ob Schaf 5 langsamer ist als Schaf 4, somit brauchen wir nur zu wissen wie sich Schaf 5 zu Schaf 4 verhält, wenn wir davon ausgehen, dass das Schaf 4 das langsamste Schaf ist, das wir bisher betrachtet haben.