Prof. Dr. Daniel Neuen
Jens Schlöter

Wintersemester 2023/2024

Algorithmentheorie

Präsenzübung 2

Präsenzübung 2.1

Zeigt oder widerlegt folgende Aussagen.

(a) $n^2 \in o(5n^3 + 2n^2 + n + 3)$.

(b) $(\log_2(n))^3 \in \mathcal{O}(n)$.

(c) $3^n \in \mathcal{O}(2^n)$.

(d) Für alle $f, g : \mathbb{N} \to \mathbb{R}$ gilt: $f \in \mathcal{O}(g)$ oder $g \in \mathcal{O}(f)$.

Lösung:

(b) Die Aussage ist wahr. Wir beobachten zuerst, dass für alle $x > 0$ gilt, dass $e^x = 1 + x + \sum_{k=2}^{\infty} \frac{x^k}{k!} > x$, somit auch $\log(x) < x$.
Dann gilt für alle $n \geq 1$:

$$
\begin{aligned}
(\log_2(n))^3 &= \left(\frac{\log(n)}{\log(2)}\right)^3 \\
&= \left(\frac{1}{\log(2)}\right)^3 \cdot (\log(\sqrt[3]{n}^3))^3 \\
&= \left(\frac{1}{\log(2)}\right)^3 \cdot 27 \cdot (\log(\sqrt[3]{n}))^3 \\
&< \frac{27}{\log(2)^3} \cdot (\sqrt[3]{n})^3 \\
&= \frac{27}{\log(2)^3} \cdot n.
\end{aligned}
$$

Wir müssen zeigen, dass es ein $c \in \mathbb{R}_+$ und ein $n_0 \in \mathbb{N}$ gibt, so dass $(\log_2(n))^3 \leq c \cdot n$ für alle $n \geq n_0$ gilt. Nach obiger Rechnung gilt diese Aussage für $n_0 = 1$ und $c = \frac{27}{\log(2)^3}$.

Präsenzübung 2.2

Gebt für folgende Rekursionsgleichungen eine geschlossene Form in Θ-Notation an und zeigt die Korrektheit per Induktion. Für beide Rekursionsgleichungen könnt ihr annehmen, dass $T(1) = 1$.

(a) $T(n) = 2T(\frac{n}{4}) + 1$.

(b) $T(n) = 3T(\frac{n}{5}) + n\log(n)$.

Lösung:

(a) $T(n) = 2T(\frac{n}{4})+1$. Wir zeigen: $T(n) \in \Theta(\sqrt{n})$. Die geschlossene Form kann mit dem ersten Fall des Mastertheorems gefunden werden.

Wir beginnen mit dem Beweis von $T(n) \in \mathcal{O}(n)$ und wählen folgende IV: $T(m) \leq c\sqrt{m}-d$ für alle $m < n$ und Konstanten c und d. Nun gilt:

$$\begin{aligned} T(n) &= 2T(\frac{n}{4}) + 1 && \text{IV} \\ &\leq 2\left(c\sqrt{n/4} - d\right) + 1 \\ &= 2c\sqrt{n/4} - 2d + 1 \\ &= c\sqrt{n} - 2d + 1 && \text{für } d \geq 1 \\ &\leq c\sqrt{n} - d. \end{aligned}$$

In unserem Induktionsschritt ist es wichtig, dass $d \geq 1$ gilt. Für den Induktionsanfang genügt es also beispielsweise für $d = 1$ und $c = n_0 = 4$ zu argumentieren:

$$T(4) = 2 \cdot 1 + 1 = 3 \leq 4 \cdot \sqrt{4} - 1 = 8 - 1 = 7.$$

Nun zeigen wir $T(n) \in \Omega(\sqrt{n})$. Hier können wir als IV einfach $T(m) \geq c\sqrt{n}$ für alle $m < n$ wählen. Dann:

$$\begin{aligned} T(n) &= 2T(\frac{n}{4}) + 1 && \text{IV} \\ &\geq 2c\sqrt{n/4} + 1 \\ &= c\sqrt{n} + 1 \\ &\geq c\sqrt{n}. \end{aligned}$$

Hier müssen wir keine Bedingungen an unser c stellen. Für den Induktionsanfang können wir einfach $c = n_0 = 1$ wählen und bekommen

$$T(1) = 1 \geq 1\sqrt{1} = 1.$$

.

(b) $T(n) = 3T\left(\frac{n}{5}\right) + n\log(n)$. Wir zeigen: $T(n) \in \Theta(n \cdot \log(n))$. Die geschlossene Form kann mit dem dritten Fall des Mastertheorems gefunden werden.

Wir zeigen zunächst $T(n) \in \mathcal{O}(n\log(n))$. Wir wählen als IV: $T(m) \leq cm\log(m)$ für alle $m < n$ und eine Konstante c. Nun gilt

$$\begin{aligned} T(n) &= 3T(\frac{n}{5}) + n\log(n) && \text{IV} \\ &\leq 3(c \cdot \frac{n}{5}\log\left(\frac{n}{5}\right)) + n\log(n) \\ &= 3(c \cdot \frac{n}{5}(\log(n) - \log(5))) + n\log(n) \\ &\leq 3(c \cdot \frac{n}{5}\log(n)) + n\log(n) \\ &= \left(\frac{3}{5}c + 1\right) n\log(n) \\ &\leq cn\log n. \end{aligned}$$

für $c \geq \frac{5}{2}$. Das heißt, wir suchen für unseren Induktionsanfang nun einen Schwellwert n_0, sodass $T(n_0) \leq \frac{5}{2} n_0 \log(n_0)$ gilt. Dies gilt zum Beispiel für $n_0 = 5$:

$$T(5) = 3 \cdot 1 + 5 \cdot \log(5) \approx 11.05 \leq \frac{5}{2} \cdot 5 \log(5) \approx 20.01.$$

Die Ω-Richtung funktioniert ähnlich. Hier können wir einfach $T(m) \geq m \log(m)$ für alle $m < n$ voraussetzen. Dann ist

$$\begin{aligned}
T(n) &= 3T(\frac{n}{5}) + n \log(n) && \text{IV} \\
&\geq \frac{3}{5} n \log(\frac{n}{5}) + n \log(n) \\
&= \frac{3}{5} n \log(n) - \frac{3}{5} \log(5) n + n \log(n) \\
&= \frac{8}{5} n \log(n) - \frac{3}{5} n \log(5) \\
&\geq n \log(n),
\end{aligned}$$

wobei die letzte Ungleichheit für alle $n \geq 5$ gilt. Das heißt, für unseren Induktionsanfang können wir einfach $n_0 = 5$ wählen (und $c = 1$ wurde einfach direkt implizit gewählt). Dann erhalten wir

$$T(5) = 3 \cdot 1 + 5 \cdot \log(5) \approx 12.05 \geq 5 \log(5) \approx 8.05.$$

Präsenzübung 2.3

Gegeben sei der Algorithmus auf der nächsten Seite, welcher als Eingabe ein Array $A = [a_0, \ldots, a_{n-1}]$ mit ganzen Zahlen bekommt. Was berechnet dieser Algorithmus? Was ist die Worst-Case Laufzeit?

Algorithmus 1 : Algorithmus für Präsenzübung 2.3

Input : Array $A = [a_0, \ldots, a_{n-1}]$ mit $a_j \in \mathbb{Z}$ für alle $j \in \{1, \ldots, n\}$.
Output : ?

$\ell = 0$
$r = n - 1$
return `divideAndConquer`(ℓ,r,A)

Function *divideAndConquer(ℓ,r,A)*
  if $\ell > r$ **then**
    return 0
  $m := \lfloor \frac{\ell + r}{2} \rfloor$
  $s_1 :=$ `divideAndConquer`(ℓ, m, A)
  $s_2 :=$ `divideAndConquer`$(m + 1, r, A)$
  $p_\ell := 0$
  $sum := 0$
  $i := m$
  while $i \geq \ell$ **do**
    $sum := sum + A[i]$
    if $sum > p_\ell$ **then** $p_\ell = sum$
    $i = i - 1$
  $p_r := 0$
  $sum := 0$
  $i := m + 1$
  while $i \leq r$ **do**
    $sum := sum + A[i]$
    if $sum > p_r$ **then** $p_r = sum$
    $i = i + 1$
  $s_3 := p_\ell + p_r$
  return $\max\{s_1, s_2, s_3\}$