Karlsruher Institut für Technologie

Algorithmen 1 – Sommersemester 2018
Zuverlässige Softwaresysteme im Kontext der Automobilindustrie
`http://verialg.iti.kit.edu`

Prof. Dr. Carsten Sinz – Markus Iser

Übungsblatt 1

Ausgabe: 25.04.2018 – 15:30
Abgabe: 02.04.2018 – 13:00

A O-Kalkül

A.1 Funktionen ordnen (3 Punkte)

1. $2^{2^{n+1}}$
2. 2^{2^n}
3. $n!$
4. e^n
5. $n \cdot 2^n$
6. 2^n
7. 1.001^n
8. $2^{\sqrt{n}}$
9. $(\log n)^{\log n}$
10. $(\log n)!$
11. n^3
12. n^2
13. $n \cdot \log n$
14. n und $2^{\log n}$
15. $\log^2 n$
16. $\log n$
17. $\log \log n$

A.2 Zusammenhang beweisen (1 Punkt)

Verwende in der Herleitung diese Zwischenschritte:

$$
\begin{aligned}
&f(n) \geq 0 \\
&g(n) \geq 0 \\
&h(n) = \max(f(n), g(n)) \\
&\qquad = \begin{cases} f(n) & \text{if } f(n) \geq g(n) \\ g(n) & \text{if } g(n) < f(n) \end{cases} \\
&f(n) \leq h(n) \\
&g(n) \leq h(n)
\end{aligned}
$$

Dann sind die Schlussfolgerungen diese:

$h(n) \in \mathcal{O}(f(n) + g(n))$ wegen $h(n) \leq c_1 \cdot (f(n) + g(n))(c_1 = 1)$

$h(n) \in \Omega(f(n) + g(n))$ wegen $2h(n) \geq f(n) + g(n) \implies h(n) \geq c_2 \cdot (f(n) + g(n))(c_2 = \frac{1}{2})$

Das gilt sogar immer, also $n_0 = 0$. Die Originalaufgabe im Cormen braucht noch Aussagen über ein n_0, da die Funktionen dort asymptotisch nicht-negativ sind. Unsere Aufgabe verwendet aber durchgängig nicht-negative Funktionen.

B Master-Theorem (1 Punkt)

$$
\begin{aligned}
\Theta(n^{log_2(7)}) &> \Theta(n^{log_4(a)}) \\
n^{log_2(7)} &> n^{log_4(a)} \\
log_2(7) &> log_4(a) \\
4^{log_2(7)} &> a
\end{aligned}
$$

Ergebnis: $a < \lfloor 4^{log_2(7)} \rfloor$

C Schleifeninvariante (4 Punkte)

Der größte gemeinsame Teiler zweier Zahlen $a, b \in \mathbb{N}_+$ ist eine Zahl $m \in \mathbb{N}_+$, für die gilt:

- m ist Teiler von a und b (in Zeichen: $m|a, m|b$)
- für alle $m' \in \mathbb{N}_+$ gilt: $m'|a \wedge m'|b \implies m'|m$

Der *Algorithmus von Euklid* berechnet zu zwei Zahlen $a, b \in \mathbb{N}_+$ deren größten gemeinsamen Teiler $m = \mathsf{gcd}(a, b)$. Algorithmus 1 beschreibt den Algorithmus in seiner Urform, wie er alten Schriften von Euklid entnommen werden kann. Eine Variante, in der die wiederholte Subtraktion durch eine Division mit Rest ersetzt wurde, finden Sie in Algorithmus 2.

Zeigen Sie für beide Varianten deren Korrektheit, indem Sie an den benötigten Stellen Assertions und Invarianten einfügen und diese beweisen. Falls notwendig, fügen Sie Hilfsvariablen ein oder annotieren Sie Variablen mit Indizes, um den entsprechenden Schleifendurchlauf zu bezeichnen.

C.1 Lösung: Euklid (original)

Algorithm 1: Euklid (original)

Data: $a, b \in \mathbb{N}_+$
Result: $\texttt{gcd}(a, b)$
while $a \neq b$ **do**
  `Helper`: $(a'', b'') \leftarrow (a, b)$
  if $a > b$ **then**
    $(a, b) \leftarrow (b, a)$
  `Invariant`: $a < b$
  `Helper`: $(a', b') \leftarrow (a, b)$
  $b \leftarrow b - a$
  `Invariant`: $\gcd(a, b) = \gcd(a, b' - a) = \gcd(a', b' - a') = \gcd(a', b') = \gcd(a'', b'')$
  `Invariant`: $a \leq a'$ und $b < b'$
return a

Wir zeigen zuerst $\gcd(a, b) = \gcd(a, b - a)$ für $a \leq b$.

Sei $m = \gcd(a, b)$ der GCD von a und b, nach Definition gilt also $m \mid a$, $m \mid b$, und für alle $m' \in \mathbb{N}_0$ mit $m' \mid a$ und $m' \mid b$ folgt $m' \mid m$. Es gibt also $z_a \in \mathbb{N}_0$ und $z_b \in \mathbb{N}_0$, so dass $a = z_a m$ und $b = z_b m$. So gilt $b - a = (z_b - z_a)m$, und daher auch $m \mid b - a$. Wegen $m \mid a$ und $m \mid b - a$ ist m also gemeinsamer Teiler von a und $b - a$.

Bleibt zu zeigen dass m *größter* gemeinsamer Teiler ist. Sei also $\tilde{m} = \gcd(a, b - a) \in \mathbb{N}_0$. Wegen $\tilde{m} \mid a$ und $\tilde{m} \mid b - a$ gilt $a = \tilde{z}_a \tilde{m}$ und $b - a = \tilde{z}_{b-a} \tilde{m}$ für zwei $\tilde{z}_a, \tilde{z}_{b-a} \in \mathbb{N}_0$, und so $b - a + a = (\tilde{z}_{b-a} + \tilde{z}_a)\tilde{m} = b$. Es gilt also $\tilde{m} \mid b$, und da auch $\tilde{m} \mid a$, folgt $\tilde{m} \mid m$ aus der Definition von m. Entsprechend folgt aus $m \mid a$ und $m \mid b - a$ auch $m \mid \tilde{m}$. Damit ist dann $m = \tilde{m}$.

Weiter gilt $\gcd(a, b) = \gcd(b, a)$, was wir hier nicht zeigen.

Mit diesen Lemmata können wir die Invarianten eintragen.

In jedem Schleifendurchlauf bleibt $\gcd(a, b)$ konstant. Der Algorithmus terminiert da $1 \leq |a + b| < |a' + b'|$, die Variablen also immer kleiner werden. Bei Abbruch ist $a = b = \gcd(a, b) = \gcd(a, a)$.

C.2 Lösung: Euklid (modulo)

Algorithm 2: Euklid (modulo)

Data: $a, b \in \mathbb{N}_+$
Result: $\texttt{gcd}(a, b)$
if $a < b$ **then**
  $(a, b) \leftarrow (b, a)$
`Assert`: $a \geq b$
while $b \neq 0$ **do**
  `Helper`: $(a', b') \leftarrow (a, b)$
  $(a, b) \leftarrow (b, a \bmod b)$
  `Invariant`: $\gcd(a, b) = \gcd(b', a' \bmod b') = \gcd(a' \bmod b', b') = \gcd(a', b')$
  `Invariant`: $a = b'$ und $b < a'$
return a

Wir zeigen zuerst $\gcd(a, b) = \gcd(a \bmod b, b)$ für $a \geq b$.

Dazu verwenden wir $\gcd(a, b) = \gcd(a - b \cdot k, b)$ für $b \cdot k \leq a$. Dies kann man einfach per Induktion aus $\gcd(a, b) = \gcd(a - b, b)$ für $a \geq b$ folgern.

Ist $a \geq b$, dann liefert Division mit Rest, ausgedrückt als $a = b \cdot (a \div b) + (a \mod b)$, die Gleichung $a \mod b = a - b \cdot (a \div b)$. Hierbei ist $a \div b := k$ eine Ganzzahl und $bk \leq a$ ist ebenfalls erfüllt. Damit kann man das vorherige Lemma auf $\gcd(a \mod b, b)$ anwenden, und erhält $\gcd(a \mod b, b) = \gcd(a - b \cdot (a \div b), b) = \gcd(a - bk, b) = \gcd(a, b)$.

$\gcd(a, b)$ bleibt in jedem Schleifendurchlauf konstant. Der Algorithmus terminiert da $1 \leq |a + b| < |a' + b'|$, die Variablen also immer kleiner werden. Bei Abbruch ist $a = \gcd(a, b) = \gcd(a, a)$.

D Karatsuba-Ofman-Algorithmus (1 Punkt)

Führen Sie den Algorithmus von *Karatsuba und Ofman* aus der Vorlesung mit den beiden Dezimalzahlen 1242 und 3163 aus. Erstellen Sie einen Berechnungsbaum, aus dem sämtliche rekursiven Aufrufe des Algorithmus ersichtlich sind.

Selber machen.