Fachbereich Mathematik & Informatik
Freie Universität Berlin
Prof. Dr. Carsten Gräser, Lasse Hinrichsen

8. Übung zur Vorlesung

Computerorientierte Mathematik I

WS 2019/2020

`http://numerik.mi.fu-berlin.de/wiki/WS_2019/CoMaI.php`

Abgabe: Freitag, 10. Januar 2019, 12:00 Uhr

1. Aufgabe (4 Bonus TP)
Sei $\mathrm{ggT}(a, b)$ der größte gemeinsame Teiler zweier positiver natürlicher Zahlen a und b. Zeigen Sie dass

$$\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a \mod b).$$

2. Aufgabe (4 TP)
Im Folgenden betrachten wir die $\mathcal{O}$- sowie die o-Notation stets für $x \to \infty$. Seien $\alpha, \beta \in \mathbb{R}$ und $g_1, g_2 : \mathbb{R} \to \mathbb{R}$ mit $g_1(x) \neq 0 \neq g_2(x)$ für alle $x \in \mathbb{R}$. Beweisen oder widerlegen Sie:

a) $x^\alpha \in \mathcal{O}\left(x^\beta\right) \Leftrightarrow \alpha \leq \beta$

b) $x^\alpha \in o\left(x^\beta\right) \Leftrightarrow \alpha \leq \beta$

c) $f_1 \in \mathcal{O}(g_1(x)) \wedge f_2 \in \mathcal{O}(g_2(x)) \Leftrightarrow f_1 + f_2 \in \mathcal{O}(|g_1(x)| + |g_2(x)|)$

d) $f_1 \in \mathcal{O}(g_1(x)) \wedge f_2 \in \mathcal{O}(g_2(x)) \Rightarrow f_1 \cdot f_2 \in \mathcal{O}(g_1(x) \cdot g_2(x))$

3. Aufgabe (4 TP)
Für $x \in \mathbb{R}$ und $n \in \mathbb{N}$ betrachten wir den folgenden Algorithmus, der hier in Form eines Pseudocodes dargestellt ist:

```
y=1
for k in range(1, n+1):
  y = y + x^k/k!
```

Hierbei werden x^k und $k!$ in jedem Schleifendurchlauf berechnet durch

$$x^k = x \cdot x \cdot \dots \cdot x, \quad k! = k \cdot (k-1) \cdot \dots \cdot 2 \cdot 1.$$

a) Beweisen Sie, dass der Aufwand dieses Algorithmus in $\mathcal{O}(n^2)$ für $n \to \infty$ ist. Mit Aufwand ist hier die Anzahl der nötigen arithmetischen Grundrechenoperationen $+, -, \cdot, /$ gemeint.

b) Geben Sie einen alternativen Algorithmus an, der denselben Wert y berechnet, dessen Aufwand aber in $\mathcal{O}(n)$ für $n \to \infty$ liegt. Beweisen Sie dies.

4. Aufgabe (8 PP + 4 Bonus TP)
Wir wollen die Laufzeiten verschiedener Algorithmen zur Bestimmung von ggT(a, b) zweier positiver natürlicher Zahlen a, b testen. Gehen Sie dafür wie folgt vor:

a) Implementieren Sie den TumbGGT-Algorithmus aus der Vorlesung als eine Funktion

`ggT_tumb(a,b)`.

Hierbei soll ein Tupel `(c,k)` zurückgegeben werden, wobei `c` der größte gemeinsame Teiler von `a` und `b` sei und `k` die Anzahl der im Algorithmus ausgeführten Divisionen sei. Eine Anwendung von `%` (`mod`) ist als einfache Division zu werten.

b) Implementieren Sie analog den Algorithmus TumbGGT++ als Funktion

`ggT_tumbpp(a,b)`

mit Rückgabepaar `(c,k)`.

c) Implementieren Sie analog den euklidischen Algorithmus als Funktion

`ggT_euclid(a,b)`

mit Rückgabepaar `(c,k)`.

d) Schreiben Sie eine Skriptdatei `run7_3.py` mit dem folgenden Ablauf:

- Sei $n = 1000$. Es werden zwei Vektoren $a, b \in \mathbb{R}^n$ mit gleichverteilten Zufallszahlen $a_i, b_i \in \{100, \cdots, 1000\}$ erstellt. Dazu können Sie beispielsweise die Funktion `numpy.random.randint` nutzen.
- Für jeden der obigen drei ggT-Algorithmen wird ein Vektor $k \in \mathbb{R}^n$ erstellt, so dass k_i die Anzahl der nötigen Divisionen ist, wenn man den jeweiligen Algorithmus für die Eingabedaten a_i und b_i anwendet.

  Im Anschluss an die Berechnung von k wird ein Histogramm für die Häufigkeit der verschiedenen k_i geplottet, in dessen Titel der Name des jeweiligen Algorithmus sowie die Werte von `kmin`$= \min_{1,\ldots,n} k_i$ und `kmax`$= \max_{1,\ldots,n} k_i$ genannt werden.

  Speichern Sie die Plots jeweils in den Dateien `hist_tumb.png`, `hist_tumbpp.png` und `hist_euclid.png` ab.

e) Geben Sie für jedes der drei Verfahren theoretische untere und obere Schranken für die Werte von $\min_i k_i$ und $\max_i k_i$ an. Beschreiben Sie außerdem Ihre Beobachtungen aus den erzeugten Daten und interpretieren Sie die Histogramme hinblicklich der theoretischen Schranken. Schreiben Sie Ihre Antwort in die Datei `beobachtungen.txt`.

Allgemeine Hinweise

Die Punkte unterteilen sich in Theoriepunkte (TP) und Programmierpunkte (PP). Bitte beachten Sie die auf der Vorlesungshomepage angegebenen Hinweise zur Bearbeitung und Abgabe der Übungszettel, insbesondere der Programmieraufgaben.