Algorithmische Mathematik:
Graphen & Anwendungen

Frühlingssemester 2018
P. Zaspel und I. Kalmykov

Universität Basel

Übungsblatt 1.

zu bearbeiten bis **Dienstag, 6.3.2018, 14:00 Uhr**.

Aufgabe 1. (Landau-Symbole)

Seien f und g Funktionen. Man sagt, dass f für $x \to a$ gegenüber g asymptotisch vernachlässigbar ist, $f = o(g)$, falls

$$\lim_{x \to a} \left| \frac{f(x)}{g(x)} \right| = 0.$$

Ausserdem heisst f für $x \to a$ asymptotisch durch g beschränkt, $f = \mathcal{O}(g)$, falls

$$\limsup_{x \to a} \left| \frac{f(x)}{g(x)} \right| < \infty.$$

Beweisen oder widerlegen Sie die folgenden Behauptungen:

a) $x^{10} + x^8 + x^6 + x^4 + x^2 = o(x^{31/3})$ für $x \to \infty$

b) $x^{10} + x^8 + x^6 + x^4 + x^2 = \mathcal{O}(x^{5/2})$ für $x \to 0$

c) $\sum_{k=0}^{n} q^k = \mathcal{O}(q^n)$ für $n \to \infty$ und alle $q > 1$

d) $\ln(x) = o(x^\alpha)$ für $x \to \infty$ und alle $\alpha > 0$

Bemerkung. Für die Bewertung der Komplexität von Algorithmen sind insbesondere Funktionen auf den natürlichen Zahlen interessant, d.h. $f \colon \mathbb{N} \to \mathbb{R}_{\geq 0}$, $g \colon \mathbb{N} \to \mathbb{R}_{\geq 0}$. Das Argument der Funktion kann hier als die Größe der Eingabe interpretiert werden. Für diese Klasse der Funktionen kann $\mathcal{O}(g)$ auch wie folgt definiert werden

$$\mathcal{O}(g) \coloneqq \{f \colon \mathbb{N} \to \mathbb{R}_{\geq 0} : \exists \alpha > 0 \; \exists n_0 \in \mathbb{N} \; \forall n \geq n_0 : f(n) \leq \alpha g(n)\}.$$

Aufgabe 2. (Bubble-sort)

Gegeben sei eine Liste von natürlichen Zahlen $S = \{s_1, \ldots, s_n\}$, d.h. $s_i \in \mathbb{N}$ für alle $i = 1, \ldots, n$. Wir möchten die Elemente von S sortieren, d.h. eine Permutation $\sigma \colon \{1, \ldots, n\} \to \{1, \ldots, n\}$ der Indizes $1, \ldots, n$ bestimmen, sodass gilt

$$\forall 1 \leq i < j \leq n \colon s_{\sigma(i)} \leq s_{\sigma(j)}.$$

Betrachten Sie dazu den folgenden Algorithmus 1.

Algorithmus 1 (Bubble-sort)

Input: Menge $S = \{s_1, \ldots, s_n\}$, $s_i \in \mathbb{N}$ für alle $i = 1, \ldots, n$
Output: Permutation $\sigma\colon \{1, \ldots, n\} \to \{1, \ldots, n\}$ mit

$$\forall 1 \leq i < j \leq n\colon s_{\sigma(i)} \leq s_{\sigma(j)}.$$

```
1: for i ← 1 to n do
2:    σ(i) ← i
3: end for
4: for i ← 1 to n − 1 do
5:    for j ← i + 1 to n do
6:       if s_σ(j) < s_σ(i) then
7:          tmp ← σ(i)
8:          σ(i) ← σ(j)
9:          σ(j) ← tmp
10:      end if
11:   end for
12: end for
13: return σ
```

a) Zeigen Sie, dass der Algorithmus 1 total korrekt ist.

   (i) Beweisen Sie die partielle Korrektheit. Benutzen Sie dabei die Invariante: *In der Iteration i ist die Liste*

   $$\left\{s_{\sigma(1)}, s_{\sigma(2)}, \ldots, s_{\sigma(i)}\right\}$$

   sortiert und jedes Element aus der Liste

   $$\left\{s_{\sigma(i+1)}, s_{\sigma(i+2)}, \ldots, s_{\sigma(n)}\right\}$$

   ist größer oder gleich als ein beliebiges Element aus

   $$\left\{s_{\sigma(1)}, s_{\sigma(2)}, \ldots, s_{\sigma(i)}\right\}.$$

   (ii) zeigen Sie, dass der Algorithmus 1 terminiert.

b) Bestimmen Sie zusätzlich die Komplexität von dem Algorithmus 1.

Aufgabe 3. (Implementierung von Bubble-sort)

Implementieren Sie den Bubble-sort Algorithmus 1. Testen Sie Ihre Implementierung an folgenden Aufgaben:

a) Sortieren einer zufällig erzeugten Menge S für ein festes $n = |S|$. $|S|$ bezeichnet dabei die Kardinalität von S.

b) In dem Algorithmus 1 betrachten wir den Vergleich in der Zeile 6 sowie die Zuweisungsoperationen in den Zeilen 7 bis 9 als elementare Rechenoperationen. Für einen Algorithmus bezeichnen wir mit Op(S) die Anzahl der elementaren Rechenoperationen für eine korrekte Eingabe S. Der mittlere Aufwand ("average case") ist die mittlere Anzahl der Operationen über alle korrekten Eingaben S. Sei $\mathcal{I}$ die Menge der korrekten Eingaben für einen Algorithmus. Dann können wir den mittleren Aufwand wie folgt berechnen

$$\frac{1}{|\mathcal{I}|} \sum_{S \in \mathcal{I}} \mathrm{Op}(S).$$

Schätzen Sie den mittleren Aufwand des Algorithmus 1 indem Sie zufällige korrekte Eingaben erzeugen und die jeweilige Anzahl der elementaren Rechenoperationen bestimmen. Benutzen Sie dabei die Stichprobengrösse von 100 bzw. 1000 Eingaben. **Hinweis.** Bitte nicht die benötigte Zeit sondern tatsächlich die Anzahl der elementaren Rechenoperationen für die Messungen benutzen.

c) Sortieren einer "best case" Eingabe, d.h. einer Menge S, sodass der Algorithmus 1 die bestmögliche Laufzeit hat. Die "best case" Eingabe ist eine bereits sortiere Liste.

d) Sortieren einer "worst case" Eingabe. Die "worst case" Eingabe ist für den Algorithmus 1 durch eine monoton absteigende Liste von Zahlen gegeben.

e) Stellen Sie für $|S| = 10, 20, \ldots, 100$ den "best case" Aufwand (d.h. die Anzahl der elementaren Operationen in c)), den "worst case" Aufwand (die Anzahl der elementaren Operationen in d)), sowie die Ergebnisse aus b) in einem Plot dar. Betrachten Sie dabei die Vergleichs- und die Zuweisungsoperationen separat.