Algorithmen und
Berechnungskomplexität I
Wintersemester 2017/18

Abgabe: 24.10.2017, 12.00 Uhr
Besprechung: KW 45

PD Dr. Elmar Langetepe
David Kübel
Institut für Informatik

Übungsblatt 2

Aufgabe 2.1: (6 Punkte)

Sei $G = \{2^{n+n}, n, n^2, n \cdot \log n, \log n, 3^{n+2}, n!, n^n\}$ mit $g : \mathbb{N} \to \mathbb{N}$ für $g \in G$. Ordnen Sie jeder nachstehenden Funktionen $f : \mathbb{N} \to \mathbb{N}$ eine Funktion $g \in G$ zu, sodass $f(n) = \Theta(g(n))$. Begründen Sie Ihre Wahl.

1. $f_1(n) = 7n^2 + 18n + 300$.
2. $f_2(n) = \log\left(\frac{n+1}{2}\right) + 7n$.
3. $f_3(n) = 7n! + 2^n$.
4. $f_4(n) = \sum_{k=0}^{n} \binom{n}{k} 2^k$.
5. $f_5(n) = n \cdot 2^{n+1} + 2^{n \cdot \log n}$.
6. $f_6(n) = \log(n^{51})$.

Hinweis: $\log n$ *steht stets für den Logarithmus von* n *zur Basis* 2.

Aufgabe 2.2: (6 Punkte)

Wir wollen die Laufzeiten von *Insertionsort* und *Mergesort* experimentell miteinander vergleichen. Erzeugen Sie dazu für $n = 1, \ldots, 500$ n-mal hintereinander ein Feld mit n zufälligen Einträgen und führen Sie *Insertionsort* und *Mergesort* auf diesem Feld aus. Bestimmen Sie für jedes n die durchschnittliche Laufzeit beider Sortierverfahren und stellen Sie diese als Graphen in Abhängigkeit von n dar.

Aufgabe 2.3: (4 Punkte)

In der Vorlesung wurde ein *Divide and Conquer*-Algorithmus zur Bestimmung des dichtesten Punktepaars in einer Menge von n Punkten in der Ebene vorgestellt. Bestimmen Sie die Laufzeit des Verfahrens mit der Substitutionsmethode.

Aufgabe 2.4: (4+4 Punkte)

Wir betrachten die Rekursionsgleichung $T(n) = 4 \cdot T(n/2) + n$ für Zweierpotenzen n mit $T(1) = 1$.

(a) Zeigen Sie, dass man die Ungleichung

$$T(n) \le cn^2 \tag{1}$$

für keine Konstante c direkt mittels Induktion zeigen kann.

(b) Beweisen Sie Ungleichung (1) für eine geeignete Konstante c, indem Sie die stärkere Ungleichung

$$T(n) \le cn^2 - f(n) \tag{2}$$

für eine geeignete Funktion $f(n) \ge 0$ induktiv zeigen.

Aufgabe 2.5: (6 Zusatzpunkte)

Sei A ein Feld mit n Einträgen $A[1], \ldots, A[n]$. Die Anzahl $\chi(A)$ der *Inversionen* oder *Fehlstellungen* von A ist definiert als

$$\chi(A) = |\{(i,j) : 1 \le i < j \le n \text{ und } A[i] > A[j]\}|.$$

Geben Sie einen Algorithmus an, der für ein Feld A mit n Einträgen die Anzahl $\chi(A)$ der Inversionen in Zeit $O(n \log n)$ bestimmt.