max planck institut informatik

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Prof. Dr. Benjamin Doerr, Dr. Reto Spöhel **Wintersemester 2011/12**
Übungsleitung: Alexander Kobel

Übung zu Grundzüge von Algorithmen und Datenstrukturen

http://www.mpi-inf.mpg.de/departments/d1/teaching/ws11/grads/

Übungsblatt 1 Abgabe: Mittwoch, 26. Oktober 2011, 17:00 Uhr

Hinweise zur Abgabe der Aufgaben: Schreiben Sie klar und deutlich Namen und Matrikelnummer sowie den Namen Ihres Tutors auf Ihre Abgabe. Heften Sie mehrere Blätter geeignet zusammen. Legen Sie ihre Lösung bis 17:00 Uhr in den Briefkasten im Erdgeschoss von Gebäude E1 3 (neben Hörsaal 001).

Hinweis zu den folgenden Aufgaben: In der MfI haben Sie bereits die O-Notation kennengelernt. Auch wenn wir diese in der Vorlesung nochmal wiederholen werden, wollen wir schon heute ein paar erste einfache Übungsaufgaben dazu stellen. Dazu genügt folgende Definition: Sei $f : \mathbb{N} \to \mathbb{R}$. Dann ist $\mathrm{O}(f) := \{g : \mathbb{N} \to \mathbb{R} \mid \exists c \in \mathbb{N}\, \exists n_0 \in \mathbb{N}\, \forall n \in \mathbb{N}_{\geq n_0} : g(n) \leq cf(n)\}$.

Aufgabe 0 (*Mündliche Übung*) Diese Aufgabe sollen Sie bis zur nächsten Übungsgruppe lösen. Dort werden Ihre Lösungen diskutiert. Sie müsssen keine schriftliche Ausarbeitung abgeben.

a) Beweisen Sie $3n^3 + 17n^2 + 68n + 101 \in \mathrm{O}(n^3)$.

b) Was müssen Sie zeigen, um $g \notin \mathrm{O}(f)$ zu beweisen?

Aufgabe 1 (*Schriftliche Übung, 4 Punkte*) Beweisen Sie die folgenden Aussagen. Stets sind $f, g, h : \mathbb{N} \to \mathbb{R}_{\geq 0}$.

a) Sei $\lambda \in \mathbb{R}_{>0}$. Wenn $f \in \mathrm{O}(g)$, dann ist auch $\lambda f \in \mathrm{O}(g)$.

b) Wenn $f_1 \in \mathrm{O}(g)$ und $f_2 \in \mathrm{O}(g)$, dann ist auch $f_1 + f_2 \in \mathrm{O}(g)$.

c) Wenn $f \in \mathrm{O}(g)$ und $g \in \mathrm{O}(h)$, dann $f \in \mathrm{O}(h)$.

Bitte wenden...

Aufgabe 2 (*Schriftliche Übung, 4 Punkte*)

a) Beweisen oder widerlegen Sie: Für alle $f, g : \mathbb{N} \to \mathbb{R}_{>0}$ gilt $f \in \mathrm{O}(g)$ oder $g \in \mathrm{O}(f)$.

b) Seien $\lambda, \mu \in \mathbb{R}_{>1}$. Zeigen Sie, dass $\log_\lambda(n) \in \mathrm{O}(\log_\mu(n))$.

Aufgabe 3 (*Schriftliche Übung, 4 Punkte*)

Geben Sie einen formalen Beweis dafür, dass der rekursive BinarySearch-Algorithmus (wie im Skript angegeben) korrekt ist und terminiert.

Aufgabe 4 (*Schriftliche Übung, 4 Punkte*)

Der Algorithmus für Binäre Suche schaut sich $\lfloor \log_2 n \rfloor + 1$ Elemente eines n-elementigen Arrays an. Man kann zeigen, dass dies optimal (im worst-case) ist. Wir wollen hier (zunächst) eine schwächere Aussage betrachten:

Sei $\mathcal{A}$ ein deterministischer Algorithmus, der für alle Eingaben $(a[1..n], x)$ korrekt entscheidet, ob x in dem sortierten Array $a[1..n]$ enthalten ist. Nehmen Sie zusätzlich an, dass $\mathcal{A}$ rein vergleichsbasiert ist, d.h., jede Entscheidung des Algorithmus ist festgelegt allein dadurch, wie sich x und die bisher betrachteten Arrayelemente vergleichen ("kleiner", "gleich" oder "größer"). Die genauen Werte von x und den Feldelementen spielen also keine Rolle.
Überlegen Sie sich, dass solche Algorithmen geeignet durch Bäume dargestellt werden können. Schließen Sie daraus, dass es eine Eingabe für $\mathcal{A}$ gibt, die mindestens $\lfloor \log_2 n \rfloor + 1$ Vergleiche benötigt.

Bonusaufgabe (*4 Bonuspunkte*)

Verzichten Sie auf die Annahme der Vergleichsbasiertheit in Aufgabe 4. Zeigen Sie also: Für jeden deterministischen Algorithmus, der für alle Eingaben $(a[1..n], x)$ korrekt entscheidet, ob x in dem sortierten Array $a[1..n]$ enthalten ist, gibt es eine Eingabe $(a[1..n], x)$, die dazu führt, dass sich der Algorithmus mindestens $\lfloor \log_2 n \rfloor + 1$ Elemente von a anschaut.