Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Dr. Hanjo Täubig
Jeremias Weihmann

Sommersemester 2012
Übungsblatt 2
30. April 2012

Grundlagen: Algorithmen und Datenstrukturen

Abgabetermin: Jeweilige Tutorübung in der Woche vom 7. bis 11. Mai

Tutoraufgabe 1

Seien $f, g : \mathbb{N}_0 \to \mathbb{R}$ Funktionen mit $\exists n_0 \in \mathbb{N} : \forall n > n_0 : f(n), g(n) > 0$. Beweisen Sie die folgenden Aussagen:

(a) $f(n) \in O(g(n))$ gilt genau dann, wenn $0 \leq \limsup_{n\to\infty} \frac{f(n)}{g(n)} < \infty$ gilt.

(b) $f(n) \in \Omega(g(n))$ gilt genau dann, wenn $0 < \liminf_{n\to\infty} \frac{f(n)}{g(n)} \leq \infty$ gilt.

(c) $f(n) \in o(g(n))$ gilt genau dann, wenn $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ gilt.

(d) $f(n) \in \omega(g(n))$ gilt genau dann, wenn $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ gilt.

(e) $f(n) \in \Theta(g(n))$ gilt genau dann, wenn $0 < \liminf_{n\to\infty} \frac{f(n)}{g(n)} \leq \limsup_{n\to\infty} \frac{f(n)}{g(n)} < \infty$ gilt.

Zur Erinnerung: Für eine Folge $h(n)$ reeller Zahlen (bzw. eine Funktion von $\mathbb{N}_0$ nach $\mathbb{R}$) gelten die folgenden Definitionen:

- $\sup M \in \mathbb{R} \cup \{-\infty, \infty\}$ ist die kleinste obere Schranke der Menge $M \subseteq \mathbb{R}$.
- $\inf M \in \mathbb{R} \cup \{-\infty, \infty\}$ ist die größte untere Schranke der Menge $M \subseteq \mathbb{R}$.
- $\limsup_{n\to\infty} h(n) := \inf_{n'\geq 0} \sup_{n\geq n'} h(n)$ ist der Limes superior (d.h. größter Häufungspunkt) von $h(n)$.
- $\liminf_{n\to\infty} h(n) := \sup_{n'\geq 0} \inf_{n\geq n'} h(n)$ ist der Limes inferior (d.h. kleinster Häufungspunkt) von $h(n)$.

Anmerkung: Falls der Grenzwert $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ existiert, dann gilt

$$\liminf_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \limsup_{n\to\infty} \frac{f(n)}{g(n)}.$$

In der Praxis ist die Existenz dieses Grenzwertes in den meisten Fällen gewährleistet, sodass viele O-Notations-Beweise auf die oftmals einfachere und elegantere Berechnung dieses Grenzwertes reduziert werden können.

Tutoraufgabe 2

Seien $f, g, h : \mathbb{N}_0 \to \mathbb{R}$ Funktionen mit $\exists n_0 \in \mathbb{N} : \forall n > n_0 : f(n), g(n), h(n) > 0$. Zeigen Sie die folgenden „Transitivitäts“regeln.

(a) Wenn $f(n) \in o(g(n))$, dann gilt $f(n) \in O(g(n))$.

(b) Wenn $f(n) \in O(g(n))$ und $g(n) \in O(h(n))$, dann gilt $f(n) \in O(h(n))$.

(c) Wenn $f(n) \in o(g(n))$ und $g(n) \in O(h(n))$, dann gilt $f(n) \in o(h(n))$.

(d) Wenn $f(n) \in O(g(n))$ und $g(n) \in o(h(n))$, dann gilt $f(n) \in o(h(n))$.

Tutoraufgabe 3

Ordnen Sie die folgenden Funktionen so an, dass für zwei in der Anordnung aufeinander folgende Funktionen f und g gilt: $f(n) \in o(g(n))$ oder $f(n) \in O(g(n))$. Beweisen Sie Ihre Anordnung. Benutzen Sie gegebenenfalls die Monotonieeigenschaften elementarer Funktionen ohne Beweis.

$$\sqrt{\operatorname{ld}(n+1)}, \quad 10^{100}, \quad \sqrt{n}, \quad \operatorname{ld}\sqrt{n+1}$$

Anmerkung: Eine solche Anordnung von Funktionen bezüglich ihres asymptotischen Wachstums und der Landau-Symbole $O, o, \Omega, \omega, \Theta$ ist im Allgemeinen nicht immer möglich, da zwei Funktionen f, g unvergleichbar sein können. Dieses Phänomen (das in der Praxis nur selten auftritt) ist Gegenstand des nächsten Übungsblattes.

Zusatzaufgabe 1

Seien $f, g, h : \mathbb{N}_0 \to \mathbb{R}$ Funktionen mit $\exists n_0 \in \mathbb{N} : \forall n > n_0 : f(n), g(n), h(n) > 0$. Zeigen Sie die folgenden „Transitivitäts“regeln.

(a) Wenn $f(n) \in \omega(g(n))$, dann gilt $f(n) \in \Omega(g(n))$.

(b) Wenn $f(n) \in \Omega(g(n))$ und $g(n) \in \Omega(h(n))$, dann gilt $f(n) \in \Omega(h(n))$.

(c) Wenn $f(n) \in \omega(g(n))$ und $g(n) \in \Omega(h(n))$, dann gilt $f(n) \in \omega(h(n))$.

(d) Wenn $f(n) \in \Omega(g(n))$ und $g(n) \in \omega(h(n))$, dann gilt $f(n) \in \omega(h(n))$.

(e) Wenn $f(n) \in \Theta(g(n))$ und $g(n) \in \Theta(h(n))$, dann gilt $f(n) \in \Theta(h(n))$.

Hausaufgabe 1

Betrachten Sie den Algorithmus 1 zur Bestimmung eines minimalen Elements in einem Integer-Array.
Bestimmen Sie die exakte Worst-Case-Laufzeit des Algorithmus, indem Sie die exakte Anzahl der Rechenoperationen, wie sie in der Vorlesung definiert wurden (Vergleiche, Zuweisungen, Additionen, . . .), in Abhängigkeit der Eingabegröße berechnen (keine asymptotische Abschätzungen durch Landau-Symbole).
Berechnen Sie nun auch die Worst-Case Laufzeit mit Hilfe der Landau-Symbole.

Algorithmus 1: Min

```
Input : int A[], int n
int i = 1
int min = A[0]
while i < n do
    if A[i] < min then
        min = A[i]
    i = i + 1
return min
```

Hausaufgabe 2

In welcher Relation bezüglich der Landau-Symbole stehen die Funktionen $\operatorname{ld} n = \log_2 n$ und $\ln n = \log_e n$? Beweisen Sie ihre Aussage.

Hausaufgabe 3

Seien $f, g : \mathbb{N}_0 \to \mathbb{R}$ Funktionen mit $\exists n_0 \in \mathbb{N} : \forall n > n_0 : f(n), g(n) > 0$.

(a) Zeigen Sie (ohne Verwendung der in den Tutoraufgaben 1 und 2 hergeleiteten Gesetze), dass $f(n) \in \Omega(g(n))$ genau dann gilt, wenn $g(n) \in O(f(n))$ gilt.

(b) Zeigen Sie (ohne Verwendung der in den Tutoraufgaben 1 und 2 hergeleiteten Gesetze), dass $f(n) \in \omega(g(n))$ genau dann gilt, wenn $g(n) \in o(f(n))$ gilt.

(c) Zeigen Sie, dass $\max\{f(n), g(n)\} \in \Theta(f(n) + g(n))$ gilt.

(d) Zeigen Sie, dass im Allgemeinen $\min\{f(n), g(n)\} \in \Theta(f(n) + g(n))$ **nicht** gilt, indem Sie geeignet gewählte Funktionen $f(n)$ und $g(n)$ als Gegenbeispiel verwenden.

Hausaufgabe 4

Ordnen Sie die folgenden Funktionen so an, dass für zwei in der Anordnung aufeinander folgende Funktionen f und g gilt: $f(n) \in o(g(n))$ oder $f(n) \in O(g(n))$. Beweisen Sie Ihre Anordnung. Benutzen Sie gegebenenfalls die Monotonieeigenschaften elementarer Funktionen ohne Beweis.

$$(\operatorname{ld} n)^{\operatorname{ld} n}, \quad n^2, \quad 2^n, \quad n, \quad \sqrt{n}, \quad n^{\operatorname{ld}\operatorname{ld} n}, \quad n \log n$$