Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
Y. Maus, O. Saukh

29 April, 2016

Algorithmen und Datenstrukturen
Sommersemester 2016
Übungsblatt 2

Abgabe bis 12:00, Freitag, 06 Mai, 2016

Aufgabe 1: (6 Punkte)

a) Beweisen Sie die folgenden Eigenschaften der Komplexitätsklassen O, Ω und Θ:

Für alle $f, g, h : \mathbb{N} \to \mathbb{R}_+$ gilt:

- Transitivität: $f \in O(g) \wedge g \in O(h) \Rightarrow f \in O(h)$, analog für Ω und Θ
- Reflexivität: $f \in O(f)$, $f \in \Omega(f)$, $f \in \Theta(f)$
- Symmetrie: $f \in \Theta(g) \Rightarrow g \in \Theta(f)$
- Antisymmetrie: $f \in O(g) \wedge g \in O(f) \Rightarrow f \in \Theta(g)$ und analog $f \in \Omega(g) \wedge g \in \Omega(f) \Rightarrow f \in \Theta(g)$

Bemerkung: Es folgt, dass Θ Äquivalenzklassen (Mengen von Funktionen) gleicher asymptotischer Komplexität definiert. O und Ω definieren Ordnungen auf Mengen von Funktionen.

b) Seien f und g zwei nicht-negative Funktionen auf $\mathbb{N}$. Beweisen Sie die folgende Aussage:

$$O(f(n)) + O(g(n)) \quad = \quad O(\max\{f(n), g(n)\})$$

Hinweis: Betrachten Sie $O(f(n))$ (bzw. $O(g(n))$, etc.) als *Menge* von *Funktionen* mit bestimmten Eigenschaften und weisen Sie nach, dass die LHS ("left hand side", die linke Seite der Gleichung) eine *Teilmenge* der RHS ("right hand side") ist *und im Fall "=" auch umgekehrt.* Betrachten Sie die Operation "+" zwischen zwei Mengen A und B wie folgt: $A + B = \{x + y | x \in A \wedge y \in B\}$. Analoges gilt für "$\cdot$".

Aufgabe 2: Komplexität von Funktionen (6 Punkte)

Das Ergebnis der vorherigen Aufgabe soll nun angewendet werden. Sortieren Sie die folgenden Funktionen nach ihrer Komplexität. Teilen Sie Ihre Liste derart in Äquivalenzklassen auf, dass f und g genau dann in derselben Äquivalenzklasse sind, wenn $f \in \Theta(g)$ gilt. Ordnen Sie danach die Komplexitätsklassen der Größe nach ($f(n) \ll g(n) \Leftrightarrow f \in O(g)$). Suchen Sie sich dann **drei** der Relationen aus und beweisen diese durch die Herleitung von Konstanten c und n_0; die übrigen Relationen müssen Sie nicht beweisen, achten Sie jedoch besonders auf die Korrektheit Ihrer Anordnung.

$$n^3,\ n^2,\ \log_3(n),\ \log_2(n),\ 2^{3+\log_2(n)},\ 2^n,\ n \cdot 2^n,\ \left(\frac{4}{3}\right)^n,\ 42,\ 2^{n+3},\ n\log_2(n),\ \sqrt{n}$$

Aufgabe 3 (8 Punkte)

Nehmen Sie an, wir haben eine Methode, um für den *QuickSort* Algorithmus in jedem Rekursionsaufruf ein Pivot-Element p zu bestimmen, so dass beim Vergleichen der Array-Elemente mit p, je mindestens $1/4$ der Werte kleiner als p und mindestens $1/4$ der Werte größer als p ist. In anderen Worten, p spaltet ein zu sortierendes Array mit m Elementen in ein Array mit λm und ein Array mit $(1-\lambda)m$ Elementen auf, wobei $\lambda \in [\frac{1}{4}, \frac{1}{2}]$.
Nehmen Sie auch an, dass es eine Konstante $b > 0$ gibt, so dass das Spalten eines Arrays der Größe m, das Auffinden des Pivot-Elements p für dieses Array und das spätere Zusammenführen der zwei sortierten Sub-Arrays, zusammen maximal bm Zeitschritte benötigt.
Beweisen Sie unter diesen Annahmen, dass für die Laufzeit $T(n)$ von *QuickSort* gilt: $T(n) \leq cn \log n$ für ein konstantes c.
Hinweis: Die Aussage lässt sich am einfachsten per Induktion beweisen. Nutzen Sie als Induktionsanfang allerdings lieber $n = 2$. Starten Sie damit, eine Ungleichung für T aufzusetzen und beachten Sie im Allgemeinen, dass λ keine Konstante ist, sondern sich bei jedem Funktionsaufruf ändern kann.