Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. Fabian Kuhn
P. Bamberger, Y. Maus, P. Schneider

Informatik 2 - Sommersemester 2018
Musterlösung Übungsblatt 2

Abgabe: Montag, 7. Mai, 14:00 Uhr

Aufgabe 1: $\mathcal{O}$-Notation *(6 Punkte)*

Beweisen oder widerlegen Sie die folgenden Aussagen. Nutzen Sie die Definitionen der $\mathcal{O}$-Notation.

(a) $100n \in \mathcal{O}(0.01n)$ *(1 Punkt)*

(b) $n \in \Omega(\log_2 3^n)$ *(1 Punkt)*

(c) $2n \in \mathcal{O}(10\sqrt{n})$ *(2 Punkte)*

(d) $f(n) + g(n) \in \Theta(\max\{f(n), g(n)\})$ für nicht negative Funktionen f und g. *(2 Punkte)*

Musterlösung

(a) Die Aussage ist korrekt. Wähle $c = 10000$ und $n_0 = 1$. Dann gilt für alle $n \geqslant n_0$

$$100n \leqslant c \cdot 0.01n.$$

(b) Die Aussage ist korrekt. Wähle $c = \log_2 3$ und $n_0 = 1$. Dann gilt für alle $n \geqslant n_0$

$$\log_2 3^n = n \cdot log_2 3 = c \cdot n.$$

(c) Die Aussage ist falsch. Sei $c > 0$. Es gilt

$$\begin{aligned} & 2n \leqslant c \cdot 10\sqrt{n} \\ \Leftrightarrow\ & n \leqslant c \cdot 5\sqrt{n} \\ \Leftrightarrow\ & \sqrt{n} \leqslant 5c \\ \Leftrightarrow\ & n \leqslant 25c^2 \end{aligned}$$

Das bedeutet, dass für jede Wahl von $c > 0$ und n_0 ein $n \geqslant n_0$ existiert mit $2n > c \cdot 10\sqrt{n}$ (wähle bspw. $n = \max\{n_0, 25c^2 + 1\}$).

(d) Es gilt $\max\{f(n), g(n)\} \leqslant f(n) + g(n) \leqslant 2 \cdot \max\{f(n), g(n)\}$

Aufgabe 2: Sortieren nach Asymptotischem Wachstum *(6 Punkte)*

Sortieren Sie folgende Funktionen nach asymptotischem Wachstum. Schreiben Sie $g <_{\mathcal{O}} f$ falls $g \in \mathcal{O}(f)$ *und* $f \notin \mathcal{O}(g)$. Schreiben Sie $g =_{\mathcal{O}} f$ falls $f \in \mathcal{O}(g)$ *und* $g \in \mathcal{O}(f)$ (Kein Beweis nötig).

n^2	$\sqrt{n}$	2^n	$\log(n^2)$
3^n	n^{100}	$\log(\sqrt{n})$	$(\log n)^2$
$\log n$	$10^{100}n$	$n!$	$n \log n$
$n \cdot 2^n$	n^n	$\sqrt{\log n}$	n

Musterlösung

	$\sqrt{\log n}$	$<_{\mathcal{O}}$	$\log(\sqrt{n})$	$=_{\mathcal{O}}$	$\log n$	$=_{\mathcal{O}}$	$\log(n^2)$
$<_{\mathcal{O}}$	$(\log n)^2$	$<_{\mathcal{O}}$	$\sqrt{n}$	$<_{\mathcal{O}}$	n	$=_{\mathcal{O}}$	$10^{100}n$
$<_{\mathcal{O}}$	$n \log n$	$<_{\mathcal{O}}$	n^2	$<_{\mathcal{O}}$	n^{100}	$<_{\mathcal{O}}$	2^n
$<_{\mathcal{O}}$	$n \cdot 2^n$	$<_{\mathcal{O}}$	3^n	$<_{\mathcal{O}}$	$n!$	$<_{\mathcal{O}}$	n^n

Aufgabe 3: Quicksort Pivotwahl *(8 Punkte)*

Wir betrachten die folgende Variante des Quicksort-Algorithmus zum Sortieren eines Arrays A der Größe n. Der Algorithmus wählt in jedem Rekursionsschritt zunächst einen Index $p \in \{0, \dots, n-1\}$, welcher die Position des Pivotelements $x := A[p]$ angibt. Im selben Rekursionsschritt werden alle Elemente in A außer x auf zwei Teilarrays L und R aufgeteilt sodass L die Elemente kleiner gleich x, und R die Elemente echt größer x enthält.

Wir gehen davon aus, dass diese Aufteilung so durchgeführt wird, dass die ursprüngliche Ordnung zwischen den Elementen innerhalb der Teilarrays L bzw. R erhalten bleibt. Schließlich wird Quicksort rekursiv auf L und R ausgeführt und die sortierten Ergebnisarrays $L, \{x\}, R$ in einem Array zurückgegeben. Im Basisfall $n \leqslant 1$ wird kein Pivot gewählt sondern das Array zurückgegeben.

Gehen Sie vereinfachend davon aus, dass jeder Rekursionsschritt in genau n Zeitschritten durchgeführt wird wobei n die Größe des *aktuell* betrachteten Arrays ist. Sie dürfen zudem $n = 2^k - 1$ für ein $k \in \mathbb{N}$ annehmen. Wir betrachten zwei Strategien zur Wahl des Pivots: (1) $p = 0$ und (2) $p = \lfloor \frac{n}{2} \rfloor$.

(a) Beschreiben Sie für beide Strategien jeweils einen Input der Länge n, der den Best-Case bzw. den Worst-case bezüglich der Gesamtlaufzeit darstellt. (ohne Beweis) *(4 Punkte)*

(b) Stellen Sie für beide Strategien (1) und (2) für jeweils Worst-case und Best-case die Rekursiongleichungen $T_{1,w}(n), T_{1,b}(n), T_{2,w}(n), T_{2,b}(n)$ für die Laufzeit in Abhängigkeit von n auf. *(2 Punkte)*

(c) Schätzen Sie die Rekursionsgleichungen aus (b) möglichst scharf nach oben ab und geben Sie die asymptotische Laufzeitklasse an. *(2 Punkte)*

Musterlösung

(a) *Worst-case $p = 0$ und Best-case $p = \lfloor \frac{n}{2} \rfloor$:* Ein aufsteigend oder absteigend sortiertes Array mit paarweise verschiedenen Werten. *(1 Punkt)*

Worst-case $p = \lfloor \frac{n}{2} \rfloor$: Bspw. ein Array welches aus den Zahlen $\{1, \dots, n\}$ besteht und in dem zunächst alle ungeraden Elemente absteigend sortiert vorkommen gefolgt von allen geraden Elementen in aufsteigender Sortierung:

$$A = (\lfloor \frac{n}{2} \rfloor, \dots, 3, 1, 2, 4, \dots, \lceil \frac{n}{2} \rceil)$$

Man beachte, dass durch die spezielle Wahl $n = 2^k - 1$ der Wert $\lfloor \frac{n}{2} \rfloor$ ungerade ist, während $\lceil \frac{n}{2} \rceil$ gerade ist. *(1 Punkt)*

Best-case $p = 0$: Wir konstruieren A aus den Zahlen $\{1, \dots, n\}$ so, dass A in jedem Rekursionsschritt in zwei genau große Teilarrays aufgeteilt wird solange bis die Arrays Länge 1 haben. Man beachte dabei dass die Annahme $n = 2^k - 1$ dies zulässt. Seien $A_1^\ell, \dots, A_m^\ell$ die Arrays die in der ℓ-ten Rekursionsstufe durch rekursives Aufteilen des ursprünglichen Arrays A entstehen. Bspw. ist $A_1^1 = A$. Seien $\mu_1^\ell, \dots, \mu_m^\ell$ die Mediane von $A_1^\ell, \dots, A_m^\ell$. Wir setzen $A[1] = \mu_1^1$. Dann setzen wir $A[2] = \mu_1^2$, $A[3] = \mu_2^2$. Des weiteren $A[3 \dots 6] = \mu_1^3, \dots, \mu_4^3$. Und so fort, d.h. alle folgenden Elemente von A sind die entsprechenden Mediane der nächsten Rekursionsstufe. *(2 Punkte)*

(b) $T_{1,w}(n) = T_{2,w}(n) = T(n-1) + n$ und $T_{1,b}(n) = T_{2,b}(n) = 2 \cdot T(\lfloor \frac{n}{2} \rfloor) + n$ mit Basisfall jeweils $T(1) = 1$. *(2 Punkte)*

(c) Es gilt $T_{1,w}(n), T_{2,w}(n) = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \implies T_{1,w}(n), T_{2,w}(n) \in \Theta(n^2)$. *(1 Punkt)*

Aus der Vorlesung: $T_{1,b}(n), T_{2,b}(n) \leqslant c \cdot n \log n \implies T_{1,b}(n), T_{2,b}(n) \in \mathcal{O}(n \log n)$ *(1 Punkt)*