Technische Universität Darmstadt
Fachgebiet Theoretische Informatik

Prof. Johannes Buchmann
Nabil Alkeilani Alkadri
Nina Bindel
Patrick Struck

Algorithmen und Datenstrukturen

SoSe 2018

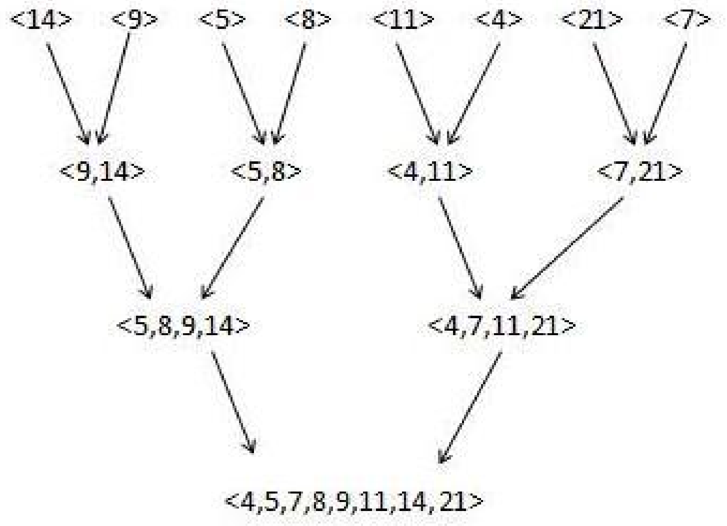
TECHNISCHE UNIVERSITÄT DARMSTADT

2. Lösungsblatt — 23.04.2018

P1 Merge-Sort

Illustrieren Sie die Operation von Merge-Sort auf dem Array $A = \langle 14, 9, 5, 8, 11, 4, 21, 7 \rangle$.

Lösung. Zunächst wird das Array A in Subarrays der Länge 1 unterteilt. Diese werden dann durch wiederholte Anwendung von Merge wie folgt zusammengesetzt.



P2 Komplexitätsklassen

1. Tragen Sie für die folgenden Funktionen die korrekten Komplexitätsklassen ein.

$f(n)$	$O(\dots)$
$5000 + 0.0001n^3 + 37n^2$	$O(\qquad)$
$30n^{1.5} + 12n \lg n$	$O(\qquad)$
$n \lg n$	$O(\qquad)$
$n \lg n^2$	$O(\qquad)$
$n^2 \lg n$	$O(\qquad)$
$n \lg n^3$	$O(\qquad)$
$n^3 \lg n$	$O(\qquad)$
$3 \log_8 n + \log_2 \log_2 \log_2 n$	$O(\qquad)$
$100n \log_3 n + n^3 + 100n$	$O(\qquad)$

Lösung.

$f(n)$	$O(\ldots)$
$5000 + 0.0001n^3 + 37n^2$	$O(n^3)$
$30n^{1.5} + 12n \lg n$	$O(n^{1.5})$
$n \lg n$	$O(n \lg n)$
$n \lg n^2$	$O(n \lg n)$
$n^2 \lg n$	$O(n^2 \lg n)$
$n \lg n^3$	$O(n \lg n)$
$n^3 \lg n$	$O(n^3 \lg n)$
$3 \log_8 n + \log_2 \log_2 \log_2 n$	$O(\lg n)$
$100n \log_3 n + n^3 + 100n$	$O(n^3)$

2. Tragen Sie für die folgenden Aussagen jeweils ein ob sie WAHR oder FALSCH sind. Falls eine Aussage falsch ist, tragen Sie die korrekte Formel ein.

Aussage	W oder F	korrekte Formel
$O(f + g) = O(f) + O(g)$		
$O(f \cdot g) = O(f) \cdot O(g)$		
Wenn $f \in O(g)$ und $h \in O(g)$, dann $f \in O(h)$		
$5n + 8n^2 + 100n^3 \in O(n^5)$		
$5n + 8n^2 + 100n^3 \in O(n^2 \lg n)$		

Lösung.

Aussage	W oder F	korrekte Formel
$O(f + g) = O(f) + O(g)$	F	$O(f + g) = \max\{O(f), O(g)\}$
$O(f \cdot g) = O(f) \cdot O(g)$	W	
Wenn $f \in O(g)$ und $h \in O(g)$, dann $f \in O(h)$	F	Wenn $f \in O(g)$ und $g \in O(h)$, dann $f \in O(h)$
$5n + 8n^2 + 100n^3 \in O(n^5)$	W	
$5n + 8n^2 + 100n^3 \in O(n^2 \lg n)$	F	$5n + 8n^2 + 100n^3 \in O(n^3)$

P3 Rekurrenzgleichung

a) Beweisen Sie die Aussage "Die Lösung von $T(n) = T(n/2) + 1$ liegt in $O(\log n)$" zunächst durch Induktion (für den Fall $n = 2^k$) und anschließend den allgemeinen Fall mit Hilfe des Mastertheorems.

Theorem 4.1 (Master theorem)

Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$T(n) = aT(n/b) + f(n)$,

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ can be bounded asymptotically as follows.

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

b) Geben Sie Beispiele für alle Fälle des Mastertheorems an und zeigen Sie, dass es sich tatsächlich um solche Fälle handelt.

Lösung.

a) 1. Sei $n = 2^k$. Wir beweisen die Aussage durch Induktion nach k.

- Induktionsanfang $k = 1$: $T(2^1) = T(2) = c_0 \leq c \cdot \lg 2$, wobei $c > 1$.
- Induktionsannahme $T(2^k) \leq c \cdot \lg 2^k$
- Induktionsschritt $k \to k+1$:

$$T(2^{k+1}) = T(2^k) + 1 \overset{(1)}{\leq} c \cdot \lg 2^k + 1 \leq c \lg 2^k + c \lg 2 = c \lg 2^{k+1}.$$

Dabei benutzen wir bei (1) die Induktionsannahme.
Damit gilt: $T(n) = O(\lg n)$ für $n = 2^k$.

2. In der Notation des Mastertheorems haben wir $a = 1$, $b = 2$ und $f(n) = 1$. Damit gilt $f(n) = \Theta(1) = \Theta(n^{\log_b a})$. Nach Teil 2 des Mastertheorems gilt also $T(n) = \Theta(n^{\log_b a} \log n) = \Theta(\log n)$.

b)
- 1. Fall: $T(n) = 4T(n/2) + 2n$. Es gilt $a = 4, b = 2$ und $f(n) = \Theta(n) = O(n^{\log_b a - \epsilon})$ für $\epsilon = 1/2$. Damit gilt $T(n) = \Theta(n^2)$.
- 2. Fall: $T(n) = 2T(n/2) + n$. Es gilt $a = b = 2$ und $f(n) = \Theta(n) = \Theta(n^{\log_b a})$. Damit gilt $T(n) = \Theta(n \log n)$.
- 3. Fall: $T(n) = 2T(n/4) + n^2$. Es gilt $a = 2$, $b = 4$ und $f(n) = \Theta(n^2) = \Omega(n^{\log_b a + \epsilon})$ mit $\epsilon = 1$. Weiterhin ist $a \cdot f(n/b) = n^2/8 \leq \frac{1}{4} f(n)$. Damit gilt $T(n) = \Theta(n^2)$.

H1 Komplexitätsklassen

Zeigen Sie die folgende Aussage:

$$o(g(n)) \cap \omega(g(n)) = \emptyset.$$

Lösung. Wir nehmen an, dass $o(g(n)) \cap \omega(g(n)) \neq \emptyset$. Das heißt, wir nehmen an, dass eine Funktion $f \in o(g(n)) \cap \omega(g(n))$ existiert. Sei nun $c > 0$ eine Konstante. Dann existiert wegen $f \in o(g(n))$ ein $n_0 \in \mathbb{N}$, sodass $0 \leq f(n) < c \cdot g(n)$ für alle $n \geq n_0$. Da auch gilt $f \in \omega(g(n))$, existiert außerdem ein $n_1 \in \mathbb{N}$, sodass $0 \leq c \cdot g(n) < f(n)$ für alle $n \geq n_1$.

Sei nun $n' = \max\{n_0, n_1\}$. Dann gilt insbesondere $f(n) < c \cdot g(n)$ und $c \cdot g(n) < f(n)$ für alle $n \geq n'$ was zu einem Widerspruch $(f(n) > f(n))$ führt. Daher war unsere Annahme falsch und es gilt

$$o(g(n)) \cap \omega(g(n)) = \emptyset.$$

H2 Komplexitätsklassen

Beweisen Sie: für zwei Funktionen f und g gilt $f(n) \in \Theta(g(n))$ genau dann wenn $f(n) \in O(g(n))$ und $f(n) \in \Omega(g(n))$.
Lösung. Um die Aussage zu beweisen, zeigen wir zwei Teilaussagen (zwei "Richtungen"):

(i) Sei $f(n) \in \Theta(g(n))$. Dann gilt $f(n) \in O(g(n))$ und $f(n) \in \Omega(g(n))$.

(ii) Sei $f(n) \in \Omega(g(n))$ und $f(n) \in O(g(n))$. Dann gilt $f(n) \in \Theta(g(n))$.

Beweis von (i):
Sei $f(n) \in \Theta(g(n))$. Nach Definition von $\Theta(g(n))$ existieren die Konstanten c_1, c_2, n_0, sodass für alle $n \geq n_0$ gilt:

$$0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n).$$

Mit c_1 und n_0 haben wir zwei Konstanten gefunden, sodass gilt $0 \leq c_1 \cdot g(n) \leq f(n)$ für alle $n \geq n_0$. Das bedeutet $f \in \Omega(g(n))$.
Gleichzeitig haben wir mit c_2 und n_0 zwei Konstanten gefunden, sodass gilt $0 \leq f(n) \leq c_2 \cdot g(n)$ für alle $n \geq n_0$. Das bedeutet $f \in O(g(n))$.

Beweis von (ii):
Sei nun $f(n) \in \Omega(g(n))$ und $f(n) \in O(g(n))$. Dann existieren Konstanten c_1, c_2, n_1, n_2 mit

$$0 \leq c_1 \cdot g(n) \leq f(n) \text{ für alle } n \geq n_1, \text{ da } f(n) \in \Omega(g(n))$$

und

$$0 \leq f(n) \leq c_2 \cdot g(n) \text{ für alle } n \geq n_2, \text{ da } f(n) \in O(g(n)).$$

Wir wählen $n_0 = \max\{n_1, n_2\}$. Damit gilt für alle $n \geq n_0$: $0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$. Das bedeutet, dass gilt $f(n) \in \Theta(g(n))$.