Zusammenfassung Informatik III

Abkürzung.
WC Worst Case
AC Worst Case
BC Best Case

Algorithmus (Insertion Sort)**.** BC: $O(n)$; AC, WC: $O(n^2)$

Notation. Sei $\mathcal{F}$ die Menge der Funktionen von $\mathbb{N}$ nach $\mathbb{R}_{\geq 0}$. Ist $g \in \mathcal{F}$, dann definieren wir

$$\begin{aligned}
O(f) &:= \{g \in \mathcal{F} \mid \exists\, c > 0\, \exists\, n_0 \in \mathbb{N}\, \forall\, n \geq n_0 : g(n) \leq c \cdot f(n)\} \\
\Omega(f) &:= \{g \in \mathcal{F} \mid \exists\, c > 0\, \exists\, n_0 \in \mathbb{N}\, \forall\, n \geq n_0 : g(n) \leq c \cdot f(n)\} \\
o(f) &:= \{g \in \mathcal{F} \mid \forall\, c > 0\, \exists\, n_0 \in \mathbb{N}\, \forall\, n \geq n_0 : g(n) \leq c \cdot f(n)\} \\
\omega(f) &:= \{g \in \mathcal{F} \mid \forall\, c > 0\, \exists\, n_0 \in \mathbb{N}\, \forall\, n \geq n_0 : g(n) \leq c \cdot f(n)\} \\
\Theta(f) &:= \{g \in \mathcal{F} \mid \exists\, c_1, c_2 > 0\, \exists\, n_0 \in \mathbb{N}\, \forall\, n \geq n_0 : \\
&\qquad c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)\} = O(f) \cap \Omega(f)
\end{aligned}$$

Satz. Seien $0 < \alpha < \beta$, $0 < a < b$ und $1 < A < B$. Betrachte

- $f_1(n) := \log \log n$
- $f_2(n) := (\log n)^\alpha$
- $f_3(n) := (\log n)^\beta$
- $f_4(n) := n^a$
- $f_5(n) := n^a (\log n)^\alpha$
- $f_6(n) := n^b (\log n)^\alpha$
- $f_7(n) := n^b$
- $f_8(n) := A^n$
- $f_9(n) := A^n \cdot n^a$
- $f_{10}(n) := A^n \cdot n^b$
- $f_{11}(n) := B^n$

Es gilt: $f_i \in o(f_{i+1})$ für $i = 1, ..., 10$.

Definition (RAM)**.** Die Random Access Access Machine besitzt eine unendlich lange Liste von aufsteigend nummerierten Speicherzellen R[0], R[1], ..., die jeweils eine ganze Zahl beinhalten und einen Programmzähler. Sie kann mittels der folgenden Sprache programmiert werden:

⟨*Zieladresse*⟩ ::= ⟨*Adresse*⟩ | R[⟨*Adresse*⟩]

⟨*Operand*⟩ ::= ⟨*Literal*⟩ | R[⟨*Adresse*⟩]

⟨*Befehl*⟩ ::= ⟨*Zieladresse*⟩ ‘`:=`’ ⟨*Operand*⟩ $\odot$ ⟨*Operand*⟩
  | ‘`if`’ ⟨*Operand*⟩ $\bowtie$ ⟨*Operand*⟩ ‘`goto`’ ⟨*Label*⟩

⟨*Programm*⟩ ::= ⟨*Befehl*⟩ ‘`;`’ ⟨*Programm*⟩ | ‘`End`’

wobei $\odot \in \{+, -, *, \div\}$ und $\bowtie \in \{<, \leq, =, \geq, >, \neq\}$. Diese einfache Grammatik lässt sich auch für unbedingte Sprünge nutzen (mittels Bedingung $0 = 0$). Ein Sprung über das Ende des Programms hinaus lässt das Programm anhalten. Per Konvention steht die Größe der Eingabe in der Speicherzelle R[1], während die tatsächliche Eingabe in R[2], ..., R[R[1] + 1] abgelegt wird.

,

Algorithmus. Zwei sortierte Folgen der Gesamtlänge n können in $O(n)$ Zeit gemischt werden.

Algorithmus (Sortieren durch Mischen / Mergesort)**.** n Elemente mit Schlüsseln aus einem total geordneten Universum können in $O(n \log n)$ Zeit nach ihren Schlüsseln sortiert werden.

Satz (Master-Theorem)**.** Seien a, b, c, k, N reelle Zahlen mit $a, c > 0$, $k \geq 0$, $b, N \in \mathbb{N}$ und $b \geq 2$ und sei $T : \mathbb{N} \to \mathbb{R}_{\geq 0}$ eine Funktion, die folgende Rekursionsungleichung erfüllt:

$$T(n) \leq \begin{cases} c, & \text{für } n \leq N \\ cn^k + aT(\lceil n/b \rceil), & \text{für } n > N \end{cases}$$

Sei ferner $\lambda := \log_b a$. Dann gilt

$$T(n) = \begin{cases} O(n^k), & \text{falls } \lambda < k \\ O(n^k \log n), & \text{falls } \lambda = k \\ O(n^\lambda), & \text{falls } \lambda > k. \end{cases}$$

Satz. Seien a, b, c, k, N reelle Zahlen mit $a, c > 0$, $k \geq 0$, $b, N \in \mathbb{N}$ und $b \geq 2$ und sei $T : \mathbb{N} \to \mathbb{R}_{\geq 0}$ eine Funktion, die folgende Rekursionsungleichung erfüllt:

$$T(n) \geq \begin{cases} c, & \text{für } n \leq N \\ cn^k + aT(\lceil n/b \rceil), & \text{für } n > N \end{cases}$$

Sei ferner $\lambda := \log_b a$. Dann gilt

$$T(n) = \begin{cases} \Omega(n^k), & \text{falls } \lambda < k \\ \Omega(n^k \log n), & \text{falls } \lambda = k \\ \Omega(n^\lambda), & \text{falls } \lambda > k. \end{cases}$$

Satz. Seien β, c, k, n reelle Zahlen mit $c, k > 0$, $n \in \mathbb{N}_0$ und $0 < \beta < 1$ und sei $T : \mathbb{N}_0 \to \mathbb{R}_{\geq 0}$ eine Funktion, die folgende Rekursionsungleichung erfüllt:

$$T(n) \leq \begin{cases} c, & \text{für } n \leq N \\ cn^k + T(\lfloor \beta n \rfloor), & \text{für } n > N. \end{cases}$$

Dann ist $T(n) = O(n^k)$

Satz (Karatsuba und Ofman)**.** Zwei n-stellige Zahlen können in $O(n^{\log_2 3})$ Zeit multipliziert werden.

Satz (Selektion)**.** Gegeben seien eine Menge X von n Elementen aus einem total geordneten Universum und eine ganze Zahl k mit $1 \leq k \leq n$. Dann können wir (deterministisch) in $O(n)$ Zeit das k-kleinste Element aus X bestimmen.