16.12.2011
Willkommen zur neunten Übung zur Vorlesung

Grundbegriffe der Informatik

Karlsruhe Institute of Technology

Matthias Janke
email: matthias.janke ät kit.edu

Organisatorisches

- Anmeldung für den Übungsschein nicht vergessen!
- Gestern waren 496 Personen angemeldet
- Da fehlen evtl immer noch ein paar Anmeldungen...
- Anmeldung über Studierendenportal: http://www.kit.edu/studieren/2873.php

Organisatorisches

- Wie gehts weiter nach Weihnachten?
- Abgabe neuntes Übungsblatt: 23.12.2011 oder früher
- Abgabe zehntes Übungsblatt: 13.1.2012 oder früher
- Übungen 10 und 11 an o.g. Terminen

Organisatorisches

- Webcast wird demnächst eingestellt
- Ab 2012 keine Übertragung in -101
- Vorlesung/Übung nur direkt im HsaF

Überblick

Aufwandsabschätzung, O-Kalkül

Scheinbar gern benutzter Merksatz:

- $g(n) \in O(f(n))$ bedeutet, dass $f(n)$ schneller wächst als $g(n)$.

O-Kalkül

Scheinbar gern benutzter Merksatz:

- $g(n) \in O(f(n))$ bedeutet, dass $f(n)$ schneller wächst als $g(n)$.
- Dies führt zu Überlegungen wie:
  n^2 wächst langsamer als $100n^2$, also gilt $100n^2 \notin O(n^2)$

O-Kalkül

Scheinbar gern benutzter Merksatz:

- $g(n) \in O(f(n))$ bedeutet, dass $f(n)$ schneller wächst als $g(n)$.
- Dies führt zu Überlegungen wie:
  n^2 wächst langsamer als $100n^2$, also gilt $100n^2 \notin O(n^2)$
- Das. Ist. Falsch.

Besser: $f(n) \in O(g(n))$ bedeutet, dass es eine Konstante c gibt, so dass $f(n)$ höchstens so schnell wächst wie $cg(n)$.

O-Kalkül

Besser: $f(n) \in O(g(n))$ bedeutet, dass es eine Konstante c gibt, so dass $f(n)$ höchstens so schnell wächst wie $cg(n)$.

- Was muss für c gelten?

O-Kalkül

Besser: $f(n) \in O(g(n))$ bedeutet, dass es eine Konstante c gibt, so dass $f(n)$ höchstens so schnell wächst wie $cg(n)$.

- Was muss für c gelten?
- Für welche n soll die Aussage gelten?

Besser: $f(n) \in O(g(n))$ bedeutet, dass es eine Konstante c gibt, so dass $f(n)$ höchstens so schnell wächst wie $cg(n)$.

- Was muss für c gelten? $c > 0$
- Für welche n soll die Aussage gelten? $\forall n \geq n_0$

O-Kalkül

Besser: $f(n) \in O(g(n))$ bedeutet, dass es eine Konstante c gibt, so dass $f(n)$ höchstens so schnell wächst wie $cg(n)$.

- Was muss für c gelten? $c > 0$
- Für welche n soll die Aussage gelten? Für "hinreichend große" n.

O-Kalkül

$$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$$

O-Kalkül

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n)) \cdot O(f_2(n))$
$\Rightarrow \exists g_1(n) \in O(f_1(n)) : \exists g_2(n) \in O(f_2(n)) :$
$g(n) = g_1(n) \cdot g_2(n)$

O-Kalkül

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n)) \cdot O(f_2(n))$
$\Rightarrow \exists g_1(n) \in O(f_1(n)) : \exists g_2(n) \in O(f_2(n)) :$
$g(n) = g_1(n) \cdot g_2(n)$
$\exists c_1, c_2 > 0 : \exists n_{01}, n_{02} \in \mathbb{N}_0 :$
$\forall i \in \{1, 2\} : \forall n \geq n_{0i} : g_i(n) \leq c_i f_i(n)$

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n)) \cdot O(f_2(n))$
$\Rightarrow \exists g_1(n) \in O(f_1(n)) : \exists g_2(n) \in O(f_2(n)) :$
$g(n) = g_1(n) \cdot g_2(n)$
$\exists c_1, c_2 > 0 : \exists n_{01}, n_{02} \in \mathbb{N}_0 :$
$\forall i \in \{1, 2\} : \forall n \geq n_{0i} : g_i(n) \leq c_i f_i(n)$
$\Rightarrow \forall n \geq \max(n_{01}, n_{02}) : g_1(n) \cdot g_2(n) \leq c_1 f_1(n) \cdot c_2 f_2(n)$
$= (c_1 c_2) f_1(n) \cdot f_2(n)$,
da alle vorkommenden Zahlen größer oder gleich 0 sind!

O-Kalkül

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n) \cdot f_2(n))$
$\Rightarrow \exists c > 0 : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : g(n) \leq cf_1(n) \cdot f_2(n)$

O-Kalkül

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n) \cdot f_2(n))$
$\Rightarrow \exists c > 0 : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : g(n) \leq cf_1(n) \cdot f_2(n)$
Wir setzen

$$g_1(n) = \begin{cases} 1 & \text{falls } n < n_0 \\ cf_1(n) & \text{sonst.} \end{cases}$$

$O(f_1(n)) \cdot O(f_2(n)) = O((f_1 \cdot f_2)(n))$
Sei $g(n) \in O(f_1(n) \cdot f_2(n))$
$\Rightarrow \exists c > 0 : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : g(n) \leq cf_1(n) \cdot f_2(n)$
Wir setzen

$$g_1(n) = \begin{cases} 1 & \text{falls } n < n_0 \\ cf_1(n) & \text{sonst.} \end{cases}$$

und

$$g_2(n) = \begin{cases} g(n)/g_1(n) & \text{falls } g_1(n) \neq 0 \\ 0 & \text{sonst.} \end{cases}$$

O-Kalkül

Es gilt: $\forall n \geq n_0 : (g_1(n) = 0 \iff cf_1(n) = 0) \land (cf_1(n) = 0 \Rightarrow g(n) \leq 0 \Rightarrow g(n) = 0)$

Es gilt: $\forall n \geq n_0 : (g_1(n) = 0 \iff cf_1(n) = 0) \wedge (cf_1(n) = 0 \Rightarrow g(n) \leq 0 \Rightarrow g(n) = 0)$
Also gilt für alle $n \in \mathbb{N}_0 : g_1(n) \cdot g_2(n) = g(n)$

Es gilt: $\forall n \geq n_0 : (g_1(n) = 0 \iff cf_1(n) = 0) \land (cf_1(n) = 0 \Rightarrow g(n) \leq 0 \Rightarrow g(n) = 0)$
Also gilt für alle $n \in \mathbb{N}_0 : g_1(n) \cdot g_2(n) = g(n)$
Außerdem gilt: $\forall n \geq n_0 : g_1(n) \leq cf_1(n) \Rightarrow g_1 \in O(f_1)$ und $\forall n \geq n_0 : g_2(n) \leq f_2(n)$.

Es gilt: $\forall n \geq n_0 : (g_1(n) = 0 \iff cf_1(n) = 0) \wedge (cf_1(n) = 0 \Rightarrow g(n) \leq 0 \Rightarrow g(n) = 0)$
Also gilt für alle $n \in \mathbb{N}_0 : g_1(n) \cdot g_2(n) = g(n)$
Außerdem gilt: $\forall n \geq n_0 : g_1(n) \leq cf_1(n) \Rightarrow g_1 \in O(f_1)$ und
$\forall n \geq n_0 : g_2(n) \leq f_2(n)$.
$g_2(n) = 0 \Rightarrow g_2(n) \leq f_2(n) \checkmark$

Es gilt: $\forall n \geq n_0 : (g_1(n) = 0 \iff cf_1(n) = 0) \wedge (cf_1(n) = 0 \Rightarrow g(n) \leq 0 \Rightarrow g(n) = 0)$
Also gilt für alle $n \in \mathbb{N}_0 : g_1(n) \cdot g_2(n) = g(n)$
Außerdem gilt: $\forall n \geq n_0 : g_1(n) \leq cf_1(n) \Rightarrow g_1 \in O(f_1)$ und $\forall n \geq n_0 : g_2(n) \leq f_2(n)$.
$g_2(n) = 0 \Rightarrow g_2(n) \leq f_2(n) \checkmark$
$g_2(n) \neq 0 \Rightarrow g_2(n) = g(n)/g_1(n)$
$\leq cf_1(n) f_2(n)/(cf_1(n)) = f_2(n) \checkmark$.

Allgemein: Es gibt eine Zahlenfolge $(n_1, n_2, \ldots)$, so dass $f(n_i)/g(n_i)$ unbegrenzt und monoton wächst
$\Rightarrow f(n) \notin O(g(n))$

Allgemein: Es gibt eine Zahlenfolge $(n_1, n_2, \ldots)$, so dass $f(n_i)/g(n_i)$ unbegrenzt und monoton wächst
$\Rightarrow f(n) \notin O(g(n))$
Wähle zu festem $c > 0$ ein $i \in \mathbb{N}_0$ so, dass $f(n_i)/g(n_i) > c$ gilt.

Allgemein: Es gibt eine Zahlenfolge $(n_1, n_2, \ldots)$, so dass $f(n_i)/g(n_i)$ unbegrenzt und monoton wächst
$\Rightarrow f(n) \notin O(g(n))$
Wähle zu festem $c > 0$ ein $i \in \mathbb{N}_0$ so, dass $f(n_i)/g(n_i) > c$ gilt.
Wähle zu festem $n_0 \in \mathbb{N}_0$ ein $j \geq i$ so dass gilt $n_j > n_0$.

Allgemein: Es gibt eine Zahlenfolge $(n_1, n_2, \ldots)$, so dass $f(n_i)/g(n_i)$ unbegrenzt und monoton wächst
$\Rightarrow f(n) \notin O(g(n))$
Wähle zu festem $c > 0$ ein $i \in \mathbb{N}_0$ so, dass $f(n_i)/g(n_i) > c$ gilt.
Wähle zu festem $n_0 \in \mathbb{N}_0$ ein $j \geq i$ so dass gilt $n_j > n_0$.
Dann gilt $cg(n_j) < (f(n_j)/g(n_j)) \cdot g(n_j) = f(n_j)$.

O-Kalkül

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Theta(n^k)$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$ und $f \in \Omega(n^k)$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

Sei $n \geq k$.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

Sei $n \geq k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \leq \prod_{i=1}^{k} \frac{n}{1} = n^k$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

Sei $n \geq k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \leq \prod_{i=1}^{k} \frac{n}{1} = n^k$

$\Rightarrow n_0 = k, c = 1$ funktioniert.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

Sei $n \geq k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \leq \prod_{i=1}^{k} \frac{n}{1} = n^k$

$f \in \{g : \mathbb{N}_0 \to \mathbb{R}_0^+ \mid \forall n \geq k : g(n) \leq 1 \cdot n^k\}$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in O(n^k)$:

Sei $n \geq k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \leq \prod_{i=1}^{k} \frac{n}{1} = n^k$

$f \in \{g : \mathbb{N}_0 \to \mathbb{R}_0^+ \mid \forall n \geq k : g(n) \leq 1 \cdot n^k\}$

$\subseteq \{g : \mathbb{N}_0 \to \mathbb{R}_0^+ \mid \exists n_0 \in \mathbb{N}_0 \exists c \in \mathbb{R}_+ :$

$\forall n \geq n_0 : g(n) \leq c \cdot n^k\} = O(n^k)$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq k$.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq k$.

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \geq \prod_{i=1}^{k} \frac{n+1-k}{k} = \frac{1}{k^k}(n+1-k)^k$$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \geq \prod_{i=1}^{k} \frac{n+1-k}{k} = \frac{1}{k^k}(n+1-k)^k$

Idee: Wenn n hinreichend groß, lässt sich $(n+1-k)$ durch $\frac{n}{2}$ abschätzen.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq 2k$.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq 2k$.

$\Rightarrow k \leq \frac{n}{2} \Rightarrow n + 1 - k \geq n + 1 - \frac{n}{2} = \frac{n}{2} + 1 \geq \frac{n}{2}$.

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq 2k$.

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \geq \prod_{i=1}^{k} \frac{n+1-k}{k} = \frac{1}{k^k}(n+1-k)^k$$

$$\geq \frac{1}{k^k}\left(\frac{n}{2}\right)^k = \frac{1}{(2k)^k} n^k$$

k fest, $f : \mathbb{N}_0 \to \mathbb{N}_0$

$$n \mapsto \begin{cases} \binom{n}{k} & \text{falls } n \geq k \\ 0 & \text{sonst} \end{cases}$$

Behauptung: $f \in \Omega(n^k)$:

Sei $n \geq 2k$.

$\binom{n}{k} = \frac{n!}{(n-k)!k!} = \prod_{i=1}^{k} \frac{n+1-i}{i} \geq \prod_{i=1}^{k} \frac{n+1-k}{k} = \frac{1}{k^k}(n+1-k)^k$

$\geq \frac{1}{k^k}(\frac{n}{2})^k = \frac{1}{(2k)^k} n^k$

$\Rightarrow n_0 = 2k$ und $c = \frac{1}{(2k)^k}$ funktionieren.

Algorithmus mit Laufzeit in $\Theta(n^3)$
Jede Woche soll der Algorithmus ein Problem lösen.
Wie groß darf Problem sein?

Algorithmus mit Laufzeit in $\Theta(n^3)$
Jede Woche soll der Algorithmus ein Problem lösen.
Wie groß darf Problem sein? $\rightarrow$ Unbekannt, m

O-Kalkül

Algorithmus mit Laufzeit in $\Theta(n^3)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner 8-mal so schnell:
Wie groß darf Problem sein?

O-Kalkül

Algorithmus mit Laufzeit in $\Theta(n^3)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner 8-mal so schnell:
Wie groß darf Problem sein? Ungefähr $2m$

Algorithmus mit Laufzeit in $\Theta(n^d)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner k-mal so schnell:
Wie groß darf Problem sein?

Algorithmus mit Laufzeit in $\Theta(n^d)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner k-mal so schnell:
Wie groß darf Problem sein? Ungefähr $\sqrt[d]{k} \cdot m$

Algorithmus mit Laufzeit in $\Theta(d^n)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner k-mal so schnell:
Wie groß darf Problem sein?

Algorithmus mit Laufzeit in $\Theta(d^n)$
Jede Woche soll der Algorithmus ein Problem lösen.
Neuer Rechner k-mal so schnell:
Wie groß darf Problem sein? Ungefähr $m + \log_d k$

Wieso muss man bei $O(\log n)$ keine Basis angeben?

Wieso muss man bei $O(\log n)$ keine Basis angeben?
$\forall a, b > 1 : \log_a n \in \Theta(\log_b(n))$

Wieso muss man bei $O(\log n)$ keine Basis angeben?

$\forall a, b > 1 : \log_a n \in \Theta(\log_b(n))$

$\log_a(n) = \frac{\log_2 n}{\log_2 a} = \frac{\log_2 b}{\log_2 b}\frac{\log_2 n}{\log_2 a} = \frac{\log_2 b}{\log_2 a}\log_b n$

Themen für das neunte Übungsblatt:

- O-Kalkül
- Äquivalenzrelationen

Schönes Wochenende!