Übung Algorithmen und Datenstrukturen

Sommersemester 2017

Patrick Schäfer, Humboldt-Universität zu Berlin

Agenda

1. Heute:
    1. Organisatorisches
    2. Vorstellung des ersten Übungsblatts
    3. Die Landau-Notation
    4. Laufzeitanalyse

Organisation

Vorlesung:	Montag	11 – 13 Uhr	Ulf Leser
	Mittwoch	11 – 13 Uhr	Ulf Leser
Übung:	Montag	13:00 - 15:00	RUD 26, 1'303, Berit Grußien
	Montag	13:00 - 15:00	RUD 26, 1'305, Marc Bux
	Dienstag	13:00 - 15:00	RUD 26, 0'313, Marc Bux
	Dienstag	13:00 - 15:00	RUD 26, 1'303, Berit Grußien
	Mittwoch	**13:00 - 15:00**	**RUD 26, 0'313, Patrick Schäfer**
	Mittwoch	**15:00 - 17:00**	**RUD 26, 0'313, Patrick Schäfer**
	Donnerstag	09:00 - 11:00	RUD 26, 0'313, Lucas Heimberg
	Freitag	09:00 - 11:00	RUD 26, 1'303, Lucas Heimberg
	Montag	13:00 - 15:00	RUD 26, 1'303, Berit Grußien
	Montag	13:00 - 15:00	RUD 26, 1'305, Marc Bux

Klausur: 7. August und 29. September 2017, jeweils 11 – 15 Uhr

Übungsschein

- 6 Übungsblätter, 50% der Punkte (150) notwendig für Übungsschein und Prüfungszulassung
- Anmeldung in Agnes
  - Keine Zulassungsgarantie zum Wunschtermin
- Anmeldung in Moodle
  - Einschreibeschlüssel: Mittwoch 13-15 Mi13.pizrm6 / Mittwoch 15-17 Mi15.sh8xbx
  - Bildung von Abgabegruppen bestehend aus zwei (nach Rücksprache drei) Studenten
  - Abgabegruppen können sich über mehrere Termine erstrecken
  - Wichtige Nachrichten werden in den Moodle-Foren angekündigt oder über Moodle versendet (ggf. Weiterleitung einrichten)
- Abgaben ohne Anmeldung in Moodle: 0 Punkte
- Abgaben mit invalider Gruppengröße: 0 Punkte
- bei vermutetem Abschreiben: 0 Punkte
- nicht ausführbare Programmieraufgaben: 0 Punkte

Übungsaufgaben

- 6 Blätter mit jeweils 50 Punkten
  - Zwei wöchentlicher Rhythmus (Moodle und auf der Website)
  - jedes Aufgabenblatt muss bearbeitet werden
  - Abgabe des ersten Aufgabenblatts: bis 8. Mai
- Schriftliche Aufgaben separat auf Papier abgeben
  - Blätter einer Aufgabe zusammentackern
  - Abgabe vor der Vorlesung bis 11:10 Uhr
  - oder bis 10:45 Uhr im Briefkasten bei Raum RUD25, 3.321
  - vorher kopieren oder abfotografieren (zur Vorstellung in der Übung)
  - jeder kommt mal dran: Freiwillige vor, ansonsten wird gelost (seid vorbereitet!)
- Programmieraufgaben (Java 8) über Moodle abgeben
  - vorher auf gruenau2 testen
- auf jedes Blatt / jede Java-Datei jeder Abgabe:
  - Namen
  - CMS-Benutzernamen
  - Moodle-Arbeitsgruppe
  - gewünschten Rückgabetermin (eindeutig: Wochentag, Uhrzeit, Dozent)

Aufgaben für Euch

- jetzt:
  - wer hat den Übungsschein bereits?
  - wer hat noch keinen Übungspartner?
- zu nächster Woche:
  - noch kein bestätigter Termin in Agnes: Für eine der Übungsgruppen 6-8 entscheiden und Berit Grußien anschreiben
  - noch kein Übungspartner: Übungspartner finden (z.B. über den dafür eingerichteten Post im Moodle-Forum)
  - in Moodle anmelden und Abgabegruppe bilden
  - dafür sorgen, dass über Agnes und Moodle versandte Nachrichten in einem regelmäßig aberufenem Postfach ankommen
  - O-Tutorial auf der Übungswebsite durchlesen
  - Grenzwerte, Ableitungen, Potenz- und Logarithmusgesetze wiederholen
  - mit der Bearbeitung von Aufgabenblatt 1 beginnen

Übungsblatt 1

Abgabe: Montag den 8.5.2017 bis 11:10 Uhr vor der Vorlesung im Hörsaal oder bis 10:45 Uhr im Briefkasten neben Raum 3.321, RUD25. Die Übungsblätter sind in Gruppen von 2 Personen zu bearbeiten. Jedes Übungsblatt muss bearbeitet werden. (Sie müssen mindestens ein Blatt für wenigstens eine Aufgabe jedes Übungsblattes abgeben.) Die Lösungen sind auf nach Aufgaben getrennten Blättern abzugeben. Heften Sie bitte die zu einer Aufgabe gehörenden Blätter vor der Abgabe zusammen. Vermerken Sie auf allen Abgaben Ihre Namen, Ihre **CMS-Benutzernamen**, Ihre Abgabegruppe (z.B. AG123) aus Moodle, und Ihren Übungstermin (z.B. Di 13 Uhr bei Marc Bux), zu dem Sie Ihre korrigierten Blätter zurückerhalten werden.
Beachten Sie auch die aktuellen Hinweise auf der Übungswebsite unter:
`https://hu.berlin/algodat17`

Konventionen:

- Für ein Array A ist $|A|$ die Länge von A, also die Anzahl der Elemente in A. Die Indizierung aller Arrays auf diesem Blatt beginnt bei 1 (und endet also bei $|A|$). Bitte beginnen Sie die Indizierung der Arrays in Ihren Lösungen auch bei 1.
- Mit der Aufforderung "Analysieren Sie die Laufzeit" ist hier gemeint, dass Sie eine möglichst gute obere Schranke der Zeitkomplexität angeben sollen und diese begründen sollen.
- Mit log wird der Logarithmus $\log_2$ zur Basis 2 bezeichnet.
- Die Menge der natürlichen Zahlen $\mathbb{N}$ enthält die Zahl 0.

Aufgabe 1 (Funktionen ordnen) **$6 \cdot 2 = 12$ Punkte**

Beweisen oder widerlegen Sie für die Teilaufgaben (a) bis (e) die folgenden Aussagen:

1. $f \in \mathcal{O}(g)$,
2. $f \in \Omega(g)$.

Hinweis: Benutzen Sie entweder die Definitionen direkt, oder betrachten Sie den Grenzwert des Quotienten der Funktionen und wenden Sie falls nötig den Satz von l'Hôpital an.

	$f(n)$	$g(n)$
(a)	$0.1n$	999^{887}
(b)	$3n^2$	$n^2 + 1000$
(c)	2^n	$n2^n$
(d)	2^n	2^{2n}

O-Notation

- Ziel: Abschätzung der Laufzeit eines Algorithmus
  - Wird definiert als Funktion der Eingabe.
  - Gesucht wird (üblicherweise) die Laufzeit im schlechtesten Fall (Worst Case).
    Das ist die Laufzeit der ungünstigsten Eingabe.
  - Die Abschätzung ist unabhängig von der Hardware oder der Implementierung.

- Grundidee: Welche Faktoren bestimmen die Laufzeit des Algorithmus, wenn die Eingabe groß wird.
  - Wir betrachten asymptotisches Wachstum der Laufzeit für $n \to \infty$
  - Intuition: nur der größte Faktor (mit dem größten Exponent) ist relevant.
    - $f(n) = 4n^4 + 5n^2 + 10 = O(n^4)$
    - D.h., n^4 dominiert für $n \to \infty$

- So erhalten wir obere oder untere Schranken der Laufzeit.

Algorithmus bar

Input: Array A der Länge $|A| = n$ mit $n \geq 2$
Output: Zahl x

```
x := 0;
for i := 1 to n do
  for j := i + 1 to n do
    if x < |A[i] − A[j]| then
      x := |A[i] − A[j]|;
    end if
  end for
end for
return x;
```

Algorithmus bar

Ordne die folgenden Funktionen bzgl. der O-Notation

a) $f_1 = \frac{1}{2} n \log n$

b) $f_2 = n^2$

c) $f_3 = \sqrt{n^3}$

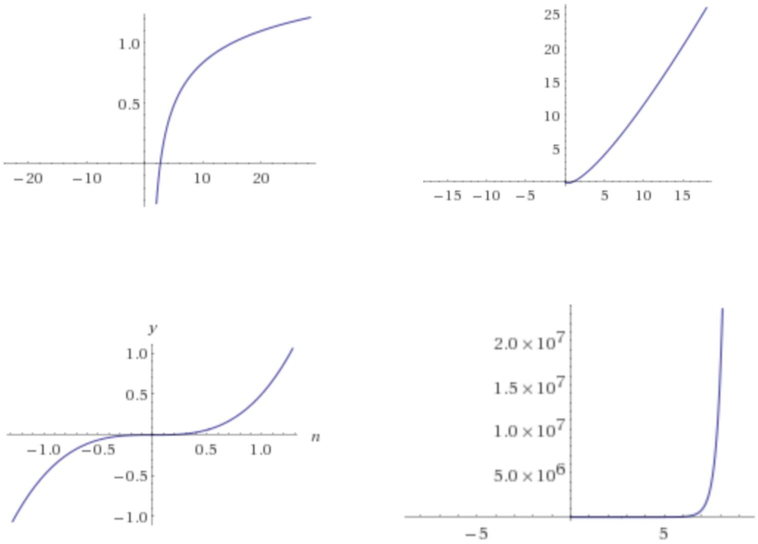
d) $f_4 = n^n$

e) $f_5 = 100n^2 + 10n^3$

f) $f_6 = 2 \log n$

g) $f_7 = n^4$

h) $f_8 = \log \log n$



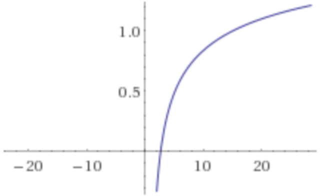
Beispiel: Lösung

Lösung: In der folgenden Liste sind die Funktionen aufsteigend bzgl. der O-Notation sortiert:

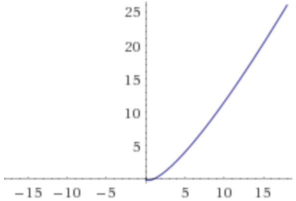
$$f_8 < f_6 < f_1 < f_3 < f_2 < f_5 < f_7 < f_4$$

bzw.

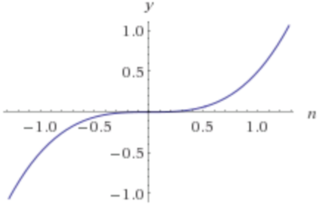
$$\log\log n < 2\log n < \frac{1}{2} n \log n < \sqrt{n^3} < n^2 < 100n^2 + 10n^3 < n^4 < n^n$$



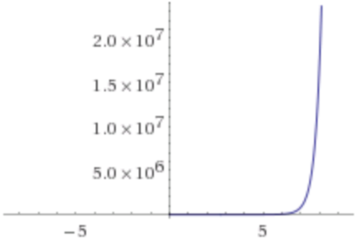
$\log\log n$



$\frac{1}{2} n \log n$



$\sqrt{n^3}$

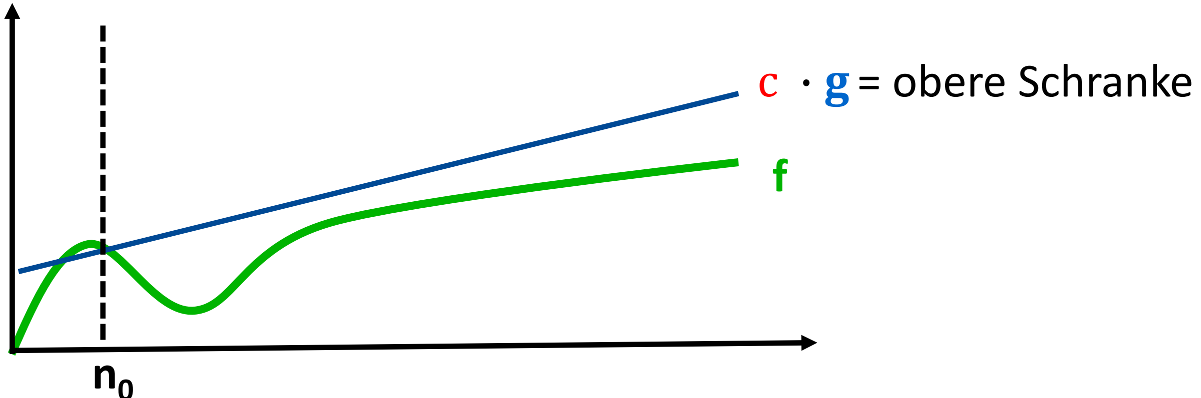


n^n

O-Notation: Menge von Funktionen

$$\underbrace{O(g)}_{a_1} = \{\underbrace{f: \mathbb{N}_0^+ \to \mathbb{R}_0^+}_{a_2} \mid \underbrace{\exists c > 0}_{a_3} \; \underbrace{\exists n_0 > 0}_{a_4} \; \underbrace{\forall n \geq n_0}_{a_5}: \; \underbrace{f(n) \leq c \cdot g(n)}_{a_6}\}$$

- a_1: Wir definieren eine Menge $O(g)$,
- a_2: Sie besteht aus der Menge aller Funktionen f,
- a_3 und a_4: Es existieren zwei Konstanten c und n_0,
- a_5: Die nachfolgende Aussage gilt ab einer Konstanten n_0,
- a_6: $c \cdot g(n)$ ist obere Schranke für $f(n)$

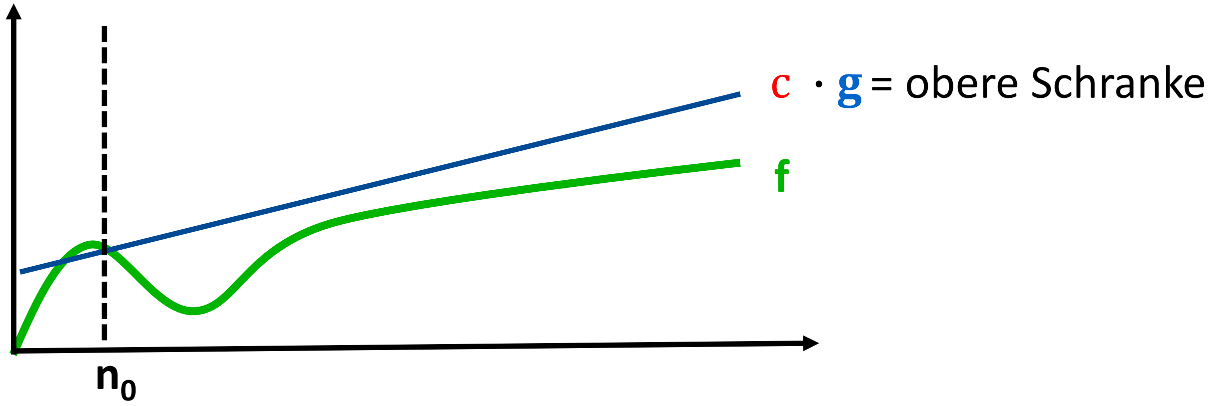


Groß O: Obere Schranke

- **Definition:**

$$O(g) = \{f\colon \mathbb{N}_0^+ \to \mathbb{R}_0^+ \mid \ \exists c > 0 \ \ \exists n_0 > 0 \ \ \forall n \geq n_0\colon \ f(n) \leq c \cdot g(n)\}$$

- Wir schreiben $f(n) \in O(g)$, falls g eine *obere Schranke* von f ist, d.h., *f wächst höchstens so schnell* wie g.



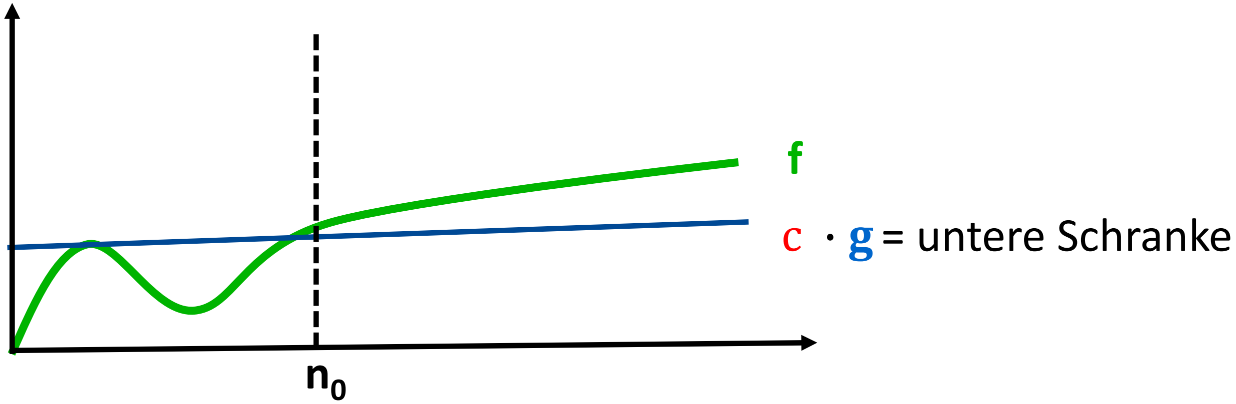
- Beispiele: $10 \cdot n^2 + n \in O(n^2)$, $10 \cdot n^2 \in O(n^3)$, $n^2 \notin O(n)$
- Scharfe obere Schranken angeben: $4n^2 - n \in O(n^2)$ sagt mehr als $4n^2 - n \in O(n^3)$.

Groß Omega: Untere Schranke

- **Definition:**

$$\Omega(g) = \{f : \mathbb{N}_0^+ \to \mathbb{R}_0^+ \mid \exists c > 0 \;\; \exists n_0 > 0 \;\; \forall n \geq n_0 : \; f(n) \geq c \cdot g(n)\}$$

- Wir schreiben $f(n) \in \Omega(g)$, falls g eine untere Schranke von f ist, d.h., f wächst *mindestens so schnell* wie g.



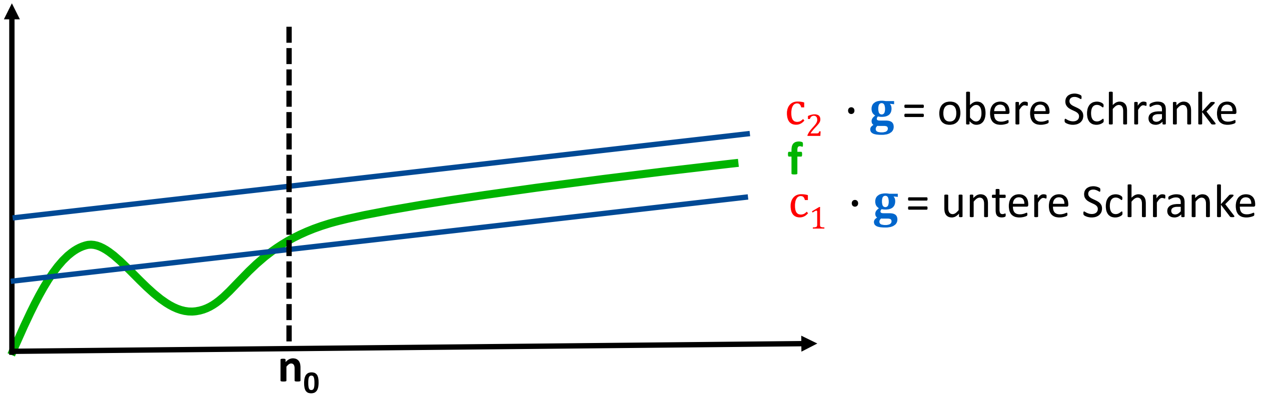
- Beispiele: $n^2 \in \Omega(\sqrt{n})$, $n^2 \in \Omega(n^2)$, $n^2 \notin \Omega(n^3)$
- Scharfe untere Schranken angeben: $4n^2 - n \in \Omega(n^2)$ sagt mehr als $4n^2 - n \in \Omega(n)$.

Theta: Asymptotisch enge Schranke

- **Definition:**

$$\Theta(g) = O(g) \cap \Omega(g)$$
$$= \{f \mid \exists c_1, c_2 > 0 \;\; \exists n_0 > 0 \;\; \forall n \geq n_0 \; : \; c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)\}$$

- Wir schreiben $f(n) \in \Theta(g)$, falls g untere und obere Schranke von f ist, d.h., f wächst *ebenso schnell* wie g.



- Beispiel: $n^2 + n \in \Theta(n^2)$

Beispiel 1, Konstante Funktionen

Hinweis: $f \in \Theta(g) \Leftrightarrow f \in \Omega(g) \quad \wedge \quad f \in O(g)$

- Gegeben zwei Funktionen, gilt $f(n) \in \Theta(g(n))$?

  $f(n) = k$ mit $k > 0$

  $g(n) = 1$

1. Zu zeigen: $k \in O(1) \Leftrightarrow (\ \exists c > 0:\ \ k \leq c \cdot 1\ )$

   wähle zB $\mathrm{c} = k$, dann gilt für alle $n \epsilon \mathbb{N}$: $k \leq c \cdot 1$

   $\Rightarrow f(n) \in O(g(n))$

2. Zu zeigen $k \in \Omega(1) \Leftrightarrow (\ \exists c' > 0:\ \ k \geq \mathrm{c}' \cdot 1\ )$

   wähle zB $\mathrm{c}' = k$, dann gilt für alle $n \epsilon \mathbb{N}$: $k \geq c' \cdot 1$

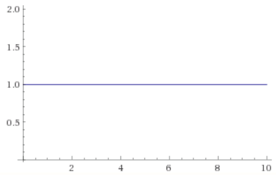
   $\Rightarrow f(n) \in \Omega(g(n))$

- Da $\mathrm{k} \in \Omega(1)$ und $\mathrm{k} \in O(1) \Longrightarrow k \in \Theta(1)$
- Allgemein: Konstante Funktionen wachsen asymptotisch gleich schnell.

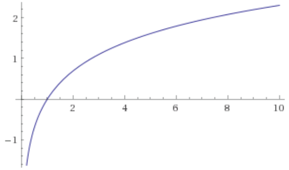
Wichtige Komplexitätsklassen

- $O(1)$: konstant (Array Zugriff)
- $O(\log n)$: logarithmisch (Binäre Suche)
- $O(n)$: linear (Sequentielle Suche)
- $O(n \log n)$: linear logarithmisch (Mergesort)
- $O(n^2)$: quadratisch (Bubble Sort)
- $O(n^k)$: polynomial
- $O(2^n)$: exponentiell (Traveling Salesman)

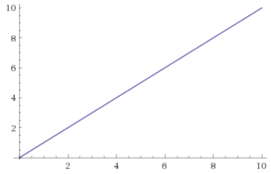
- Komplexität bis zu $O(n^2)$ ist akzeptabel.



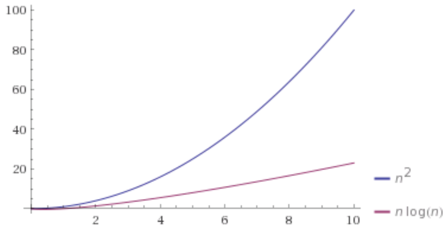
konstant



logarithmisch

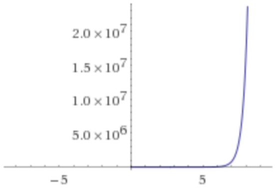


linear



linear logarithmisch

polynomial



exponentiell

Die Landau-Notation

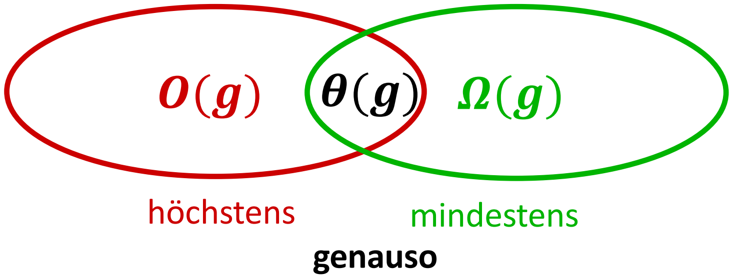
- höchstens so schnell wie...
- $\color{red}{O(g)} = \{f \mid \exists c > 0 \;\; \exists n_0 > 0 \;\; \forall n \geq n_0\colon \;\; f(n) \color{red}{\leq} c \cdot g(n)\}$

- mindestens so schnell wie...
- $\color{green}{\Omega(g)} = \{f \mid \exists c > 0 \;\; \exists n_0 > 0 \;\; \forall n \geq n_0\colon \;\; f(n) \color{green}{\geq} c \cdot g(n)\}$

- genauso schnell wie ...
- $\mathbf{\Theta(g)} = O(g) \cap \; \Omega(g)$

- wesentlich langsamer als ...
- $o(g) = \{f \mid \color{red}{\forall c > 0} \;\; \exists n_0 > 0 \;\; \forall n \geq n_0\colon \;\; f(n) \color{red}{<} c \cdot g(n)\}$

- wesentlich schneller als ...
- $\omega(g) = \{f \mid \color{red}{\forall c > 0} \;\; \exists n_0 > 0 \;\; \forall n \geq n_0\colon \;\; f(n) \color{red}{>} c \cdot g(n)\}$



O-Notation - Zusammenhänge

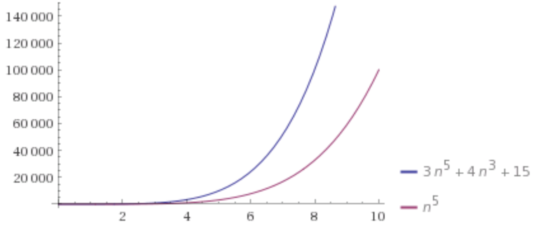
	$\mathcal{O}$	Ω	Θ	o	ω
$f \in o(g)$	X	—	—	X	—
$f \in \Theta(g)$	X	X	X	—	—
$f \in \omega(g)$	—	X	—	—	X

- $f \in O(g) \Leftrightarrow g \in \Omega(f)$
- $f \in o(g) \Leftrightarrow g \in \omega(f)$
- $f \in o(g) \Longrightarrow f \notin \Omega(g) \quad \wedge \quad f \in O(g)$
- $f \in \omega(g) \Longrightarrow f \notin O(g) \quad \wedge \quad f \in \Omega(g)$

Beispiel 2, Polynome

- Gegeben zwei Funktionen, gilt $f(n) \in \Theta(g(n))$?
  - $f(n) = 3n^5 + 4n^3 + 15$
  - $g(n) = n^5$

Hinweis: $f \in \Theta(g) \Leftrightarrow f \in \Omega(g) \quad \wedge \quad f \in O(g)$



1. Zu zeigen: $f(n) \in O(g(n)) \Leftrightarrow (\exists c, n_0 > 0 \;\; \forall n \geq n_0\colon\; 3n^5 + 4n^3 + 15 \leq c \cdot n^5)$

   Wähle zB $c = 3 + 4 + 15 = 22$,

   dann gilt $\forall n \geq 1$: $3n^5 + 4n^3 + 15 \leq 3n^5 + 4n^5 + 15n^5 \leq 22n^5$

   $\Rightarrow f(n) \in O(n^5)$

2. Zu zeigen: $f(n) \in \Omega(g(n)) \Leftrightarrow (\exists c', n_0 > 0 \;\; \forall n \geq n_0\colon\; 3n^5 + 4n^3 + 15 \geq c' \cdot n^5)$

   Wähle zB $c' = 3$,

   dann gilt $\forall n \geq 1$: $3n^5 + 4n^3 + 15 \geq 3n^5$

   $\Rightarrow f(n) \in \Omega(n^5)$

$\Rightarrow f(n) \in \Theta(g(n))$

Allgemeine Beweise, Exponentialfunktionen

- Seien $a, b \in \mathbb{R}$ positive Konstanten, dann gilt:

$a^{x+b} \in \Theta(a^x)$ z.B. $3^{x+2} \in \Theta(3^x)$ Hinweis: $f \in \Theta(g) \Leftrightarrow f \in \Omega(g) \wedge f \in O(g)$

1) Zu zeigen: $a^{x+b} \in O(a^x) \Leftrightarrow \left(\exists c, n_0 > 0 \forall x \geq n_0: \ a^{x+b} \leq c \cdot a^x\right)$

wähle $c = a^b$, dann gilt $\forall x \geq 1$: $a^{x+b} = a^x \cdot a^b = c \cdot a^x$

$\Rightarrow a^{x+b} \in O(a^x)$

2) Zu zeigen: $a^x \in O(a^{x+b}) \Leftrightarrow \left(\exists c', n_0 > 0 \forall x \geq n_0: a^x \leq c' \cdot a^{x+b}\right)$

wähle $c' = \frac{1}{a^b}$, dann gilt $\forall x \geq 1$: $a^x = a^x \cdot \frac{a^b}{a^b} = c' \cdot a^b \cdot a^x = c' \cdot a^{x+b}$

$\Rightarrow a^x \in O(a^{x+b})$

$\Rightarrow a^x \in \Theta(a^{x+b})$

Grenzwert als hinreichendes Kriterium

- Satz: $\lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty \Rightarrow f \in O(g)$ $\Leftrightarrow g \in \Omega(f)$
  - D.h. Funktion im Nenner wächst *mindestens so schnell* wie im Zähler
  - Beispiel: $\frac{n^2}{n^3}$, $\frac{n^3}{1.5\,n^3}$
- Satz: $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Leftrightarrow f \in o(g)$ $\Leftrightarrow g \in \omega(f)$
  - D.h. Funktion im Nenner wächst *schneller*
  - Beispiel: $\frac{n^2}{n^3}$

- Satz: $\lim_{n\to\infty} \frac{f(n)}{g(n)} > 0 \Rightarrow f \in \Omega(g)$ $\Leftrightarrow g \in O(f)$
  - D.h. Funktion im Nenner wächst *höchstens so schnell*
  - Beispiel: $\frac{1.5\,n^3}{n^3}$, $\frac{n^3}{n^2}$
- Satz: $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \Leftrightarrow f \in \omega(g)$ $\Leftrightarrow g \in o(f)$
  - D.h. Funktion im Nenner wächst *langsamer*
  - Beispiel: $\frac{n^3}{n^2}$

Beispiel 1, analog zum Übungsblatt

- Gilt $100n \in O(n^2)$ oder$100n \in \Omega(n^2)$?

Zu zeigen: $100n \in o(n^2)$

Hinweis: $f \in o(g) \Longrightarrow \; f \notin \Omega(g) \quad \wedge \quad f \in O(g)$

Wir betrachten den Grenzwert:

$$\lim_{n\to\infty} \frac{100n}{n^2} = \lim_{n\to\infty} \frac{100}{n} = 0$$

$\Longrightarrow 100n \in \mathrm{o}(n^2)$
$\Longrightarrow 100n \in O(n^2)\; und 100n \notin \Omega(n^2)$

- Alternativ kann auch $n^2 \in \omega(100n)$ gezeigt werden.

Beispiel 2, analog zum Übungsblatt

- Gilt $f(n) \in O\big(g(n)\big)$ oder $f(n) \in \Omega\big(g(n)\big)$?

  $f(n) = 3n^5 + 4n^3 + 15$

  $g(n) = n^5$

1. Zu zeigen: $f(n) \in O\big(g(n)\big)$

   Wir betrachten den Grenzwert:

   $$\lim_{n\to\infty} \frac{3n^5 + 4n^3 + 15}{n^5} = \lim_{n\to\infty} \left(\frac{3n^5}{n^5} + \frac{4n^3}{n^5} + \frac{15}{n^5}\right) = 3 + 0 + 0 = 3$$

   $\Longrightarrow f(n) \in O\big(g(n)\big)$

2. Zu zeigen: $f(n) \in \Omega\big(g(n)\big)$

   Wir betrachten den Grenzwert:

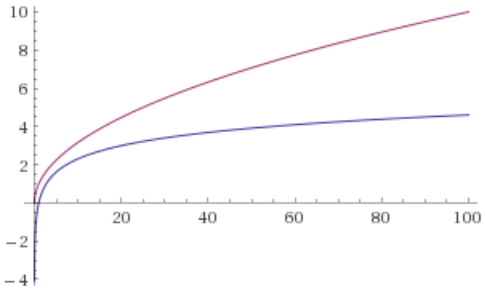
   $$\lim_{n\to\infty} \frac{n^5}{3n^5 + 4n^3 + 15} = \lim_{n\to\infty} \frac{n^5}{n^5\left(3 + \frac{4}{n^2} + \frac{15}{n^5}\right)} = \lim_{n\to\infty} \frac{1}{\left(3 + \frac{4}{n^2} + \frac{15}{n^5}\right)} = \frac{1}{3}$$

   $\Longrightarrow f(n) \in \Omega\big(g(n)\big)$

=> $f(n) \in \Theta\big(g(n)\big)$

Beispiel 3, analog zum Übungsblatt

- Gilt $f(n) \in O(g(n))$ oder $f(n) \in \Omega(g(n))$?
  - $f(n) = \log n \ (= \log_2 n)$
  - $g(n) = \sqrt{n}$



1. Zu zeigen: $f(n) \in o(g(n))$

   Wir betrachten den Grenzwert:

   $$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} = \frac{\infty}{\infty} = \cdots$$

Satz von L‘Hôpital

- Satz von L’Hôpital:
  Seien f und g zwei differenzierbare Funktionen, deren Grenzwerte entweder beide gegen 0 oder beide gegen ∞ gehen. Dann gilt :

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \overset{L'H}{=} \lim_{n\to\infty} \frac{f'(n)}{g'(n)}$$

(falls der Grenzwert existiert).

Logarithmengesetze

Basen transformieren:

$$log_a n = \frac{log_b a \;\cdot\; log_a n}{log_b a} = \frac{log_b \left(a^{log_a n}\right)}{log_b a} = \frac{log_b n}{log_b a}$$

Beispiel:

$$log_2 n = \frac{\log_e n}{\log_e 2} = \frac{\ln n}{\ln 2}$$

Wofür brauchen wir das? Zur Ableitungen von Logarithmen:

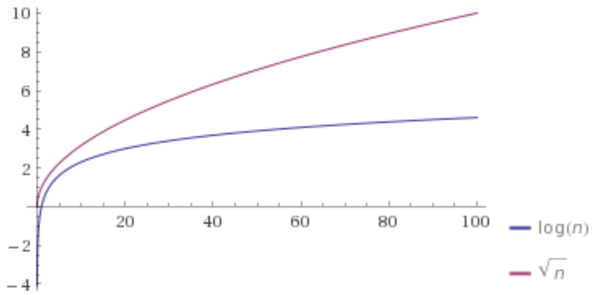
$$(\ln n)' = \frac{1}{n}$$

Analog:

$$(\log_2 n)' = \left(\frac{\ln n}{\ln 2}\right)' = \frac{1}{\ln 2 \cdot n}$$

Beispiel 3, mit L‘Hôpital

- Gilt $f(n) \in O\big(g(n)\big)$ oder $f(n) \in \Omega\big(g(n)\big)$?
  - $f(n) = \log n \; (= \log_2 n)$
  - $g(n) = \sqrt{n}$



1. Zu zeigen: $f(n) \in o\big(g(n)\big)$

   Wir betrachten den Grenzwert:

$$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} = \lim_{n\to\infty} \frac{\ln n}{\ln 2 \cdot \sqrt{n}}$$

$$\overset{L'H}{=} \lim_{n\to\infty} \frac{(\ln n)'}{(\ln 2 \cdot \sqrt{n})'}$$

$$= \lim_{n\to\infty} \frac{2}{n \cdot \ln 2 \cdot n^{-\frac{1}{2}}} = \lim_{n\to\infty} \frac{2}{\ln 2 \cdot \sqrt{n}} = 0$$

$\Rightarrow f \in o(g),\ f \notin \Omega(g)$

Ausblick

- zu nächster Woche:
  - noch kein bestätigter Termin in Agnes: Für eine der Übungsgruppen 6-8 entscheiden und Berit Grußien anschreiben
  - noch kein Übungspartner: Übungspartner finden (z.B. über den dafür eingerichteten Post im Moodle-Forum)
  - in Moodle anmelden und Abgabegruppe bilden
  - dafür sorgen, dass über Agnes und Moodle versandte Nachrichten gelesen werden
  - O-Tutorial auf der Übungswebsite durchlesen
  - Grenzwerte, Ableitungen, Potenz- und Logarithmusgesetze wiederholen
  - mit der Bearbeitung von Aufgabenblatt 1 beginnen
- nächste Woche:
  - Wiederholung und Vertiefung der O-Notation
  - Vorbereitung Aufgabenblatt 1, Aufgaben 3 & 4 (Pseudocode-Analyse, Algorithmenentwurf)
  - Klärung von Fragen zu Aufgabenblatt 1

Allgemeine Beweise: Transitivität

- Zeigen Sie, dass für beliebige Funktionen $f, g, h : \mathbb{N} \to \mathbb{N}$ gilt:

  $f \epsilon O(g) \wedge g \epsilon O(h) \Rightarrow f \epsilon O(h)$

- Gegeben:

  $\exists c_1, n_1\colon \forall n \geq n_1\colon f(n) \leq c_1 \cdot g(n) \quad \wedge \quad \exists c_2, n_2\colon \forall n \geq n_2\colon g(n) \leq c_2 \cdot h(n)$

- Zu zeigen:

  $\exists c_3, n_3\colon \forall n \geq n_3\colon f(n) \leq c_3 \cdot h(n)$

- Wähle $c_3 = c_1 \cdot c_2$ und $n_3 = \max(n_1, n_2)$, dann gilt:

  $\forall n \geq n_3\colon f(n) \leq c_1 \cdot g(n) \leq c_1 \cdot c_2 \cdot h(n) \leq c_3 \cdot h(n)$

$\Rightarrow f \epsilon O(h)$

Komplexeres Beispiel

- Gilt $f(n) \in O\big(g(n)\big)$ oder $f(n) \in \Omega\big(g(n)\big)$?
  - $f(n) = \log^2 n = \log n \cdot \log n$
  - $g(n) = \sqrt{n}$

Z.z.: $f(n) \in o\big(g(n)\big)$

Wir betrachten den Grenzwert:

$$\lim_{n \to \infty} \frac{\log^2 n}{\sqrt{n}} = \cdots$$

Weitere Beispiele

- $n \, log_2 n \in O(n^2)$

Beispiel 3 mit L'Hôpital

- Beispiel:
  - $f(n) = 3n^5 + 4n^3 + 15$
  - $g(n) = n^5$
- Gilt $f(n) \in \Omega(g(n))$?
- Wir betrachten den Grenzwert:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{n^5}{3n^5 + 4n^3 + 15}$$

$$\overset{L'H}{=} \lim_{n\to\infty} \frac{(n^5)'}{(3n^5+4n^3+15)'} = \lim_{n\to\infty} \frac{5n^4}{5\cdot 3n^4 + 3\cdot 4n^2}$$

$$\overset{L'H}{=} \lim_{n\to\infty} \frac{4\cdot 5n^3}{5\cdot 3n^4 + 2\cdot 3\cdot 4n}$$

$$\overset{L'H}{=} \lim_{n\to\infty} \frac{4\cdot 5n^3}{4\cdot 5\cdot 3n^3 + 2\cdot 3\cdot 4\cdot 1}$$

$$\overset{L'H}{=} \lim_{n\to\infty} \frac{3\cdot 4\cdot 5n^2}{3\cdot 4\cdot 5\cdot 3n^2} = \lim_{n\to\infty} \frac{1}{3} = \frac{1}{3}$$

$\Rightarrow f \in \Omega(g)$