Informatik II: Algorithmen und Datenstrukturen
SS 2015

Vorlesung 3, Dienstag, 5. Mai 2015
(O-Notation)

Prof. Dr. Hannah Bast
Lehrstuhl für Algorithmen und Datenstrukturen
Institut für Informatik
Universität Freiburg

Blick über die Vorlesung heute

- Organisatorisches
  - Ihre Erfahrungen mit dem Ü2 (Induktion + Laufzeit QuickSort)
  - Korrekturen von Ihrem Tutor: wie und wann
  - Morgen Fragestunde im **HS 051-03-026**
- O-Notation
  - Grundlagen: Motivation, Definition, Beispiele
  - Grenzwert: Bestimmung über den $\lim_{n \to \infty}$
  - Diskussion: Nutzen und Grenzen
  - Übungsblatt 3: Die Laufzeit von ein paar Ausdrücken / Programmen als $\Theta(\ldots)$ bestimmen

Morgen (Mittwoch) keine Vorlesung

- Morgen (Mittwoch) keine Vorlesung
  - Morgen (Mittwoch) keine Vorlesung
  - Stattdessen gibt es eine **Fragestunde**, und zwar im

    Raum 051-03-026 (Gebäude 51 unter dem Dach)

    Mit Kandelblick (auf der linken Seite)
  - Zwei Teile:

    Allgemeine Fragen zur Vorlesung / zum Drumherum

    Spezielle Fragen zu Ihrer Entwicklungsumgebung etc.

    Deswegen auch nicht im HS 026, weil man da nicht gut zu einzelnen Leuten an den Platz gehen kann

Erfahrungen mit dem Ü2 1/3

oder $\{0, \dots, n-1\}$

- Zusammenfassung / Auszüge Stand 5. Mai 12:00
  - Für die meisten weniger Arbeit als das Ü1
  - Sehr große Unterschiede bei den Bearbeitungszeiten
  - Einige bei Aufgabe 3 Probleme mit Linearzeit

    Was zeigt, dass das eine gute + wichtige Aufgabe war
  - Einige Rückfragen, was denn eine Permutation sei

    Eine Abbildung der Menge {1, …, n} auf sich selber
  - Arbeitsaufwand variiert stark zwischen Aufgaben

    Ja, aber variiert auch stark zwischen Studierenden
  - Wo sind die Extrafragen vom letzten Jahr (C++)

Erfahrungen mit dem Ü2   2/3

- Zusammenfassung / Auszüge   Fortsetzung
  - Unklarheiten bei Aufgabenstellung

    Immer gleich im Forum fragen !
  - Ab und zu eine Fragestunde wäre gut

    Sie haben es so gewollt
  - Mehr Background dazu, wie man richtig Mathe macht

    Ich schaue mal, was sich tun lässt
  - "Keine Zeit gehabt wegen verlängertem Wochenende"

Erfahrungen mit dem Ü2   3/3

- Lösungsskizzen … Details siehe Musterlösung

A1 : zu zeigen : $a^{\log_b n} = n^{\log_b a}$

$$a^{\log_b n} = a^{\frac{\ln n}{\ln b}} = a^{\frac{\ln n}{\ln a} \cdot \frac{\ln a}{\ln b}}$$

$$= a^{\log_a n \cdot \log_b a} = n^{\log_b a}$$

A2 : morgen

A3 : Beispiel : $\overset{3}{2}, \overset{4}{3}, \overset{5}{3}, \overset{1}{1}, \overset{2}{1}$ ⇐ Permutation = gewünschte Ausgabe

5 ← Zwischensummen

(1) $\#0 = \overset{0}{0}, \#1 = \overset{2}{2}, \#2 = \overset{3}{1}, \#3 = 2$

ab 0   ~~ab 0~~ ab 1   ab 2   ~~ab 3~~ ab 4

2, 3, 4, 0, 1 ← wie gewünscht, nur −1 (Beginn bei 0)

ALTERNATIVE : $2, \underset{4}{3}, 3, {}_{2}\,1, 1$

$$\begin{array}{ccccc} \hline \multicolumn{1}{|c|}{-1} & \multicolumn{1}{c|}{3} & \multicolumn{1}{c|}{0} & \multicolumn{1}{c|}{1} & \multicolumn{1}{c|}{-1} \\ \hline 0 & 1 & 2 & 3 & 4 \end{array} \rightarrow \dot{3}, \dot{4}, \dot{0}, \dot{1}, \dot{2} \qquad \begin{array}{ccccc} \hline \multicolumn{1}{|c|}{2} & \multicolumn{1}{c|}{3} & \multicolumn{1}{c|}{4} & \multicolumn{1}{c|}{0} & \multicolumn{1}{c|}{1} \\ \hline 0 & 1 & 2 & 3 & 4 \end{array}$$

Feedback von Ihrem Tutor

- Das funktioniert so
  - Im SVN in einer Datei im zum Ü gehörigen Ordner, z.B.

    xy123/uebungsblatt-01/feedback-tutor.txt

    Machen Sie einfach **svn update** in Ihrer Arbeitskopie
  - Einigen hatten um Korrektur vor der Abgabe des nächsten Übungsblattes gebeten

    Korrektur in der Regel bis Montag Nachmittag

    Falls Sie mit dem Zeitpunkt der Korrektur unzufrieden sind, schreiben Sie Ihrem Tutor eine Mail und einigen Sie sich

    Hat in der Vergangenheit immer sehr gut funktioniert so

O-Notation – Grundlagen 1/10

- Erinnerung
  - Wir haben jetzt mehrfach die Laufzeit $T(n)$ in Abhängigkeit von der Eingabegröße abgeschätzt
  - Die Werte der Konstanten waren dabei sekundär … und auch, wenn die Schranken erst ab $n \geq$ irgendeinem n_0 galten

    Für sehr kleine Eingaben sind Programme ja sowieso schnell
  - Zum Beispiel war beim Sortieren interessant:

    Die Laufzeit von MinSort "wächst wie" n^2

    Die Laufzeit von QuickSort "wächst wie" $n \cdot \log n$

    Vergleichsbasiertes Sortieren "geht nicht schneller als" $n \cdot \log n$

O-Notation – Grundlagen 2/10

- Motivation
  - Genau das wollen wir jetzt formaler machen, damit wir in Zukunft präzise sagen bzw. schreiben können

    Die Laufzeit von MinSort ist **$\Theta(n^2)$**

    Die Laufzeit von QuickSort ist **$O(n \cdot \log n)$**

    Die Laufzeit von CountingSort ist **$O(n)$**

    Vergleichsbasiertes Sortieren hat Laufzeit **$\Omega(n \cdot \log n)$**

O-Notation – Grundlagen 3/10

- Vorbetrachtung
  - Wir betrachten Funktionen $f : \mathbf{N} \rightarrow \mathbf{R}$

    N = die natürlichen Zahlen ... typisch: Eingabegröße

    R = die reellen Zahlen ... typisch: Laufzeit

    Uns reicht, wenn $f(n) > 0$ für $n \geq n_0$ … darunter darf f negativ sein, und das kommt bei Abschätzungen auch manchmal raus
  - Beispiele

    $f(n) = 3 \cdot n + 3$

    $f(n) = 2 \cdot n \cdot (\log_2 n - 5)$ < 0 für $n = 1, 2, \ldots, 31$

    $f(n) = n^2 / 10$

    $f(n) = n^2 + 3 \cdot n \cdot \log_2 n - 4 \cdot n$

O-Notation – Grundlagen 4/10

eigentlich $g \in O(f)$

- Groß-O, Definition
  - Seien g und f zwei Funktionen **N** → **R**
  - **Intuitiv:** Man sagt g ist Groß-O von f ...

    wenn g "höchstens so stark wächst wie" f
  - **Informal:** Man schreibt $g = O(f)$ ...

    wenn ab irgendeinem Wert n_0 für all $n \geq n_0$

    $g(n) \leq C \cdot f(n)$ für irgendeine Konstante C
  - **Formal:** für eine Funktion $f : \mathbf{N} \to \mathbf{R}$ ist ...

    $$O(f) = \{ g : \mathbf{N} \to \mathbf{R} \mid \exists n_0 \in \mathbf{N} \;\; \exists C>0 \;\; \forall n \geq n_0 \;\; g(n) \leq C \cdot f(n) \}$$

    dabei heißt ∃ = "es existiert ..." und ∀ = "für alle ..."

O-Notation – Grundlagen 5/10

- Groß-O, Beispiel
  - Sei $g(n) = 5 \cdot n + 7$ und $f(n) = n$
  - Dann ist $g = O(f)$ bzw. man schreibt $5 \cdot n + 7 = O(n)$
  - **Intuitiv:** $5 \cdot n + 7$ wächst höchstens "linear"
  - Beweis unter Verwendung der Definition von O :

Zu zeigen: $\exists n_0 \; \exists C \; \forall n \ge n_0 : g(n) \le \underline{C \cdot f(n)}$

$$g(n) = 5 \cdot n + \underbrace{7}_{\le n \text{ für } n \ge \underbrace{7}_{=:n_0}} \le 5 \cdot n + n = \underbrace{6}_{=:C} \cdot n = C \cdot f(n) \quad \blacksquare$$

ALTERNATIVE:

$$g(n) = 5 \cdot n + \underbrace{7}_{\le 7 \cdot n \text{ für } n \ge \underbrace{1}_{=:n_0}} \le 5 \cdot n + 7 \cdot n = \underbrace{12}_{=:C} \cdot n = C \cdot f(n) \quad \blacksquare$$

O-Notation – Grundlagen 6/10

- Es zählt "Wachstumsrate", nicht absolute Werte
  - Für zwei Funktionen kann ohne Probleme gelten

    $g = O(f)$ g wächst **nicht stärker** als f

    $g > f$ g ist überall **echt größer** als f
  - Zum Beispiel g und f von der Folie vorher

$g(n) = 5 \cdot n + 7 , \; f(n) = n$

$5 \cdot n + 7 > n \; \forall n$!!!

O-Notation – Grundlagen 7/10

- Groß-Omega, Definition + Beispiel
  - **Intuitiv:** Man sagt g ist Groß-Omega von f ...

    ... wenn g "mindestens so stark wächst wie" f

    Also wie Groß-O, nur mit "mindestens" statt "höchstens"
  - **Formal:** Für eine Funktion $f : \mathbf{N} \rightarrow \mathbf{R}$ ist

    $\Omega(f) = \{ g : \mathbf{N} \rightarrow \mathbf{R} \mid \exists n_0 \in \mathbf{N} \ \ \exists C > 0 \ \ \forall n \geq n_0 \ \ g(n) \geq C \cdot f(n) \}$
  - Zum Beispiel $5 \cdot n + 7 = \Omega(n)$
  - Beweis unter Verwendung der Definition von Ω :

$$5 \cdot n + \underbrace{7}_{\geq 0 \ \forall n} \geq \underbrace{5}_{=:C} \cdot n \qquad \blacksquare$$

O-Notation – Grundlagen 8/10

- Groß-Theta, Definition + Beispiel
  - **Intuitiv:** Man sagt g ist Theta von f ...

    ... wenn g "genauso so stark wächst wie" f
  - **Formal:** Für eine Funktion $f : \mathbf{N} \to \mathbf{R}$ ist

    $\Theta(f) = O(f) \cap \Omega(f)$ = die Schnittmenge von $O(f)$ und $\Omega(f)$

    Wächst "höchstens so stark" **und** "mindestens so stark"
  - Zum Beispiel $5 \cdot n + 7 = \Theta(n)$
  - Beweis unter Verwendung der Definition von Θ :

Folie 11 : $5 \cdot n + 7 = O(n)$

Folie 13 : $5 \cdot n + 7 = \Omega(n)$

$\Rightarrow 5 \cdot n + 7 = \Theta(n)$ ∎

O-Notation – Grundlagen 9/10

- Es gibt auch noch o (Klein-O) und ω (Klein-Omega)
  - Die braucht man in der Informatik viel seltener
  - Hier kurz die Definitionen für $f : \mathbf{N} \rightarrow \mathbf{R}$

    $o(f) = \{ g : \forall \varepsilon > 0 \ \ \exists n_0 \in \mathbf{N} \ \ \forall n \geq n_0 \ \ g(n) < \varepsilon \cdot f(n) \}$

    $\omega(f) = \{ g : \forall C > 0 \ \ \exists n_0 \in \mathbf{N} \ \ \forall n \geq n_0 \ \ g(n) > C \cdot f(n) \}$

  - Intuitiv:

    $g = o(f)$: g wächst (strikt) langsamer als f

    $g = \omega(f)$: g wächst (strikt) schneller als f

    Insbesondere ist die Schnittmenge leer: $o(f) \cap \omega(f) = \emptyset$

O-Notation – Grundlagen 10/10

- Intuitive Zusammenfassung
  - Die Operatoren O , Ω , Θ , o , ω sind auf Funktionen, was die Operatoren ≤ , ≥ , = , < , > auf Zahlen sind:

    O entspricht ≤

    Ω entspricht ≥

    Θ entspricht =

    o entspricht <

    ω entspricht >
  - Viele Eigenschaften übertragen sich auch

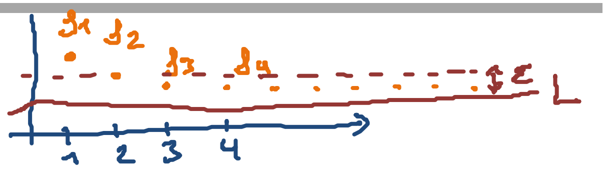
    $x < y \wedge y \leq z \Rightarrow x < z$

    z.B. Transitivität: $f = o(g) \wedge g = O(h) \Rightarrow f = o(h)$

    z.B. Additivität: $f_1 = O(g_1) \wedge f_2 = O(g_2) \Rightarrow f_1+f_2 = O(g_1+g_2)$

    $x_1 \leq y_1 \quad x_2 \leq y_2 \Rightarrow x_1+x_2 \leq y_1+y_2$

O-Notation – Grenzwerte 1/7



- Grenzwertbegriff
  - Die Definitionen auf den vorherigen Folien erinnern sehr stark an den **Grenzwertbegriff** aus der **Analysis**
  - **Definition:** Eine unendliche Folge $f_1, f_2, f_3, \ldots$ hat einen Grenzwert L, wenn für alle $\varepsilon > 0$ ein $n_0 \in \mathbf{N}$ existiert so dass für alle $n \geq n_0$ gilt dass $|f_n - L| \leq \varepsilon$
  - In Symbolen schreibt man dann $\lim_{n\to\infty} f_n = L$
  - Eine Funktion $f : \mathbf{N} \to \mathbf{R}$ kann man genauso gut als Folge $f(1), f(2), f(3), \ldots$ auffassen und schreibt $\lim_{n\to\infty} f(n) = L$

O-Notation – Grenzwerte 2/7

- Beispiel für einen Beweis von einem Grenzwert
  (sollten Sie eigentlich in Mathe 1 schon mal gesehen haben)

$$\lim_{n\to\infty} \frac{1}{n} = 0$$

Zu zeigen $\forall \varepsilon > 0\ \exists n_0\ \forall n \ge n_0 : \left|\frac{1}{n} - \overset{=L}{0}\right| \le \varepsilon$

Sei $\varepsilon > 0$ beliebig (z.B. $\frac{1}{1000}$)

$! =$ hätte ich gerne

$$\left|\frac{1}{n} - 0\right| = \frac{1}{n} \overset{!}{\le} \varepsilon$$ gilt für $n \ge \frac{1}{\varepsilon}$

also $n_0 := \left\lceil \frac{1}{\varepsilon} \right\rceil$

$$\forall n \ge n_0 : \frac{1}{n} \underset{\substack{\uparrow \\ n \ge n_0}}{\le} \frac{1}{n_0} = \frac{1}{\left\lceil \frac{1}{\varepsilon} \right\rceil} \underset{\lceil x \rceil \ge x}{\le} \frac{1}{1/\varepsilon} = \varepsilon \quad \blacksquare$$

O-Notation – Grenzwerte 3/7

- Satz
  - Seien $f, g : \mathbf{N} \rightarrow \mathbf{R}$ und der Grenzwert $\lim_{n\rightarrow\infty} f(n)/g(n)$ existiert (evtl. ist er ∞)
  - Dann gelten

(1) $f = O(g) \Leftrightarrow \lim_{n\rightarrow\infty} f(n)/g(n) < \infty$

(2) $f = \Omega(g) \Leftrightarrow \lim_{n\rightarrow\infty} f(n)/g(n) > 0$

(3) $f = \Theta(g) \Leftrightarrow \lim_{n\rightarrow\infty} f(n)/g(n) > 0$ und $< \infty$

(4) $f = o(g) \Leftrightarrow \lim_{n\rightarrow\infty} f(n)/g(n) = 0$

(5) $f = \omega(g) \Leftrightarrow \lim_{n\rightarrow\infty} f(n)/g(n) = \infty$

O-Notation – Grenzwerte 4/7

- Beweis von (1) ... die anderen Beweise gehen analog

Zu zeigen: $f = O(g) \Leftrightarrow \lim\limits_{n \to \infty} f(n)/g(n) < \infty$

„⇒": $f = O(g) \Rightarrow \exists C\, \exists n_0\, \forall n \geq n_0 : \underbrace{f(n) \leq C \cdot g(n)}$

Annahme: $\lim\limits_{n \to \infty} f(n)/g(n) = \infty$

$\Rightarrow \exists n_0\, \forall n \geq n_0 \quad \frac{f(n)}{g(n)} \geq C + 1$ (zum Beispiel)

$\Rightarrow \forall n \geq n_0 \quad f(n) \geq (C+1) \cdot g(n) > C \cdot g(n)$ ↯

„⇐": $\lim\limits_{n \to \infty} \frac{f(n)}{g(n)} = C < \infty \Rightarrow \varepsilon = 1, \exists n_0\, \forall n \geq n_0 \quad \left| \frac{f(n)}{g(n)} - C \right| \leq 1$

$\Rightarrow \frac{f(n)}{g(n)} \leq C + 1$

$\Rightarrow f(n) \leq (C+1) \cdot g(n) \quad \forall n \geq n_0$

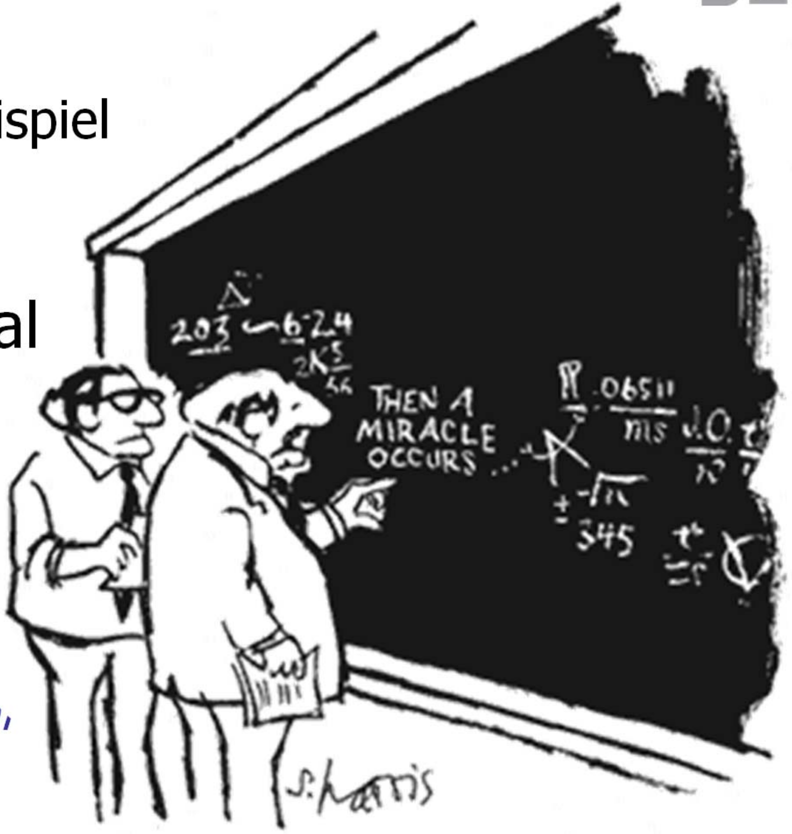
$\Rightarrow f = O(g)$ ∎

O-Notation – Grenzwerte 5/7

- Variante 1: "zu Fuß"
  - Dafür hatten wir gerade das Beispiel

    $\lim_{n\to\infty} 1/n = 0$
- Variante 2: Regel von L'Hôpital
  - Seien $f, g : \mathbf{N} \to \mathbf{R}$ wie gehabt
  - Es existieren die ersten Ableit
    Grenzwert $\lim_{n\to\infty} f'(n)/g'(n)$

    $\lim_{n\to\infty} f(n)/g(n) = \lim_{n\to\infty} f'(n$
- Variante 3: "sieht man doch"
  - Erst mit Professur erlaubt …



"I THINK YOU SHOULD BE MORE EXPLICIT HERE IN STEP TWO."

O-Notation – Grenzwerte 6/7

- Beispiel: Grenzwert mit L'Hôpital

z.B. $\lim\limits_{n \to \infty} \frac{\ln n}{n} \quad \begin{matrix} \to \infty \\ \to \infty \end{matrix} \quad \frac{\infty}{\infty} = ?$

$f(n) = \ln n \quad , \quad g(n) = n$

$f'(n) = \frac{1}{n} \quad , \quad g'(n) = 1$

$\lim\limits_{n \to \infty} \frac{1/n}{1} = \lim\limits_{n \to \infty} \frac{1}{n} = 0$

$\Longrightarrow \lim\limits_{n \to \infty} \frac{\ln n}{n} = \lim\limits_{n \to \infty} \frac{1/n}{1} = 0 \quad \blacksquare$

O-Notation – Grenzwerte 7/7

- Was darf man ohne Beweis annehmen?
  - Gute Frage !!! Da gibt es keine klare Regel

    Im Zweifelsfall immer mehr beweisen als weniger
  - **Beispiel 1:** $\lim_{n \to \infty} 1/n = 0$

    Brauchen Sie nicht mehr weiter zu beweisen
  - **Beispiel 2:** $\lim_{n \to \infty} 1/n^2 = 0$

    Einfach auf so was wie Beispiel 1 zurückführen
  - **Beispiel 3:** $\lim_{n \to \infty} (\log n)/n = 0$

    Hier ist ein Argument angebracht, z.B. mit L'Hôpital

O-Notation – Diskussion 1/2

- Sprechweise
  - Die O-Notation schaut sich das Verhalten der Funktionen an, wenn $n \to \infty$ geht (es interessieren nur die $n \geq n_0$)
  - Wenn man Laufzeiten o.ä. als $O(\ldots)$, $\Omega(\ldots)$, $\Theta(\ldots)$, $o(\ldots)$ oder $\omega(\ldots)$ ausdrückt, spricht man daher von **asymptotischer Analyse**
- Vorsicht
  - Asymptotische Analyse sagt nichts über das Laufzeitverhalten bei "kleinen" Eingabegrößen ($n < n_0$) aus
  - Für $n < 2$ oder $n < 10$ ist das egal, da wird schon nichts Schlimmes passieren
  - Aber das n_0 ist nicht immer so klein ... nächste Folie

O-Notation – Diskussion 2/2

- Beispiel
  - Algorithmus A hat Laufzeit $f(n) = 80 \cdot n$ „linear"
  - Algorithmus B hat Laufzeit $g(n) = 2 \cdot n \cdot \log_2 n$ „$n \cdot \log n$"
  - Dann ist $f = O(g)$ und sogar $f = o(g)$

    Insbesondere für alle $n \geq$ irgendein n_0 : $f(n) \leq g(n)$

    Das heißt, A ist asymptotisch echt schneller als B
  - Allerdings:

$$n_0 = 2^{40} = (2^{10})^4 \approx 1000^4 \approx 10^{12} = 1 \text{ Billionen } \overset{\text{(TERA)}}{}$$

$$\forall n < n_0 : \quad g(n) = 2 \cdot n \cdot \underbrace{\log_2 n}_{<40}$$

$$< 80 \cdot n = f(n)$$

Literatur / Links

- O-Notation / Ω-Notation / Θ-Notation
  - In Mehlhorn/Sanders:

    2.1 Asymptotic Notation
  - In Wikipedia

    http://en.wikipedia.org/wiki/Big_O_notation

    http://de.wikipedia.org/wiki/Landau-Symbole