Algorithmen und Datenstrukturen (für ESE) WS 2011 / 2012

Vorlesung 2, Dienstag, 8. November 2011
(Asymptotische Analyse / O-Notation)

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

Blick über die Vorlesung heute

- Organisatorisches
  - Übungsgruppen, Einteilung und Termine
  - Ihre Erfahrungen mit dem 1. Übungsblatt
- Asymptotische Analyse / O-Notation
  - Motivation
  - Definition von O, Ω, Θ
  - Beispiele
  - Bezug zu Grenzwerten
  - Vorsicht

Übungsgruppen

- Auf Ihrer Daphne Seite
  - ... sehen Sie, welchem Tutor Sie zugewiesen sind
    - bei Wechsel-Wunsch, bitte Mail an Björn Buchhold
  - Übungsgruppen pro Tutor (siehe auch Wiki)
    - Sebastian Sester Dienstags 14:15 Uhr
    - Katja Faist Mittwochs 12:15 Uhr
    - Manuel Bührer Donnerstags 12:15 Uhr
  - Finden statt ab nächster Woche (15./16./17. November)
    - Raum wird noch angekündigt

Ihre Erfahrungen mit dem 1. Ü-Blatt

- Zusammenfassung von Ihrem Feedback Stand 8.11. 16:00
  - Drumherum hat viel Zeit gekostet (Ant, SVN, ...)
    - SVN und Ant wurden nicht richtig erklärt
  - Zu schwierig und/oder zu viel Arbeit (bis zu 20h für einige)
    - fehlende Programmierpraxis / viel vergessen
    - fehlende Erfahrung im Umgang mit Beweisen
  - Die meisten haben um die 10h gebraucht (was ok ist !)
  - Wechsel von C++ nach Java hat Zeit gekostet
    - Sie können es auch in C++ machen, wenn Sie möchten!
  - Videoaufzeichnung mit 150% Geschwindigkeit geguckt
  - Hilfe im Forum war sehr schnell und gut
    - Hinweise zu Aufgabe 3 Teil 1 nicht hilfreich
  - "Aller Anfang ist schwer"

MergeSort — Laufzeitanalyse

- Wiederholung + Nachlese
  - Wir hatten gezeigt $T(n) \leq n \cdot T(1) + A \cdot n \cdot \log_2 n$ für eine nicht näher spezifizierte Konstante $A > 0$
  - Sei A' eine Konstante die $\geq A$ und $\geq T(1)$
    - man beachte, dass $T(1)$ auch eine Konstante ist
  - Dann ist $T(n) \leq A' \cdot n + A' \cdot n \cdot \log_2 n = A' \cdot n \cdot (1 + \log_2 n)$
  - Für $n \geq 2$ ist das $\leq 2A' \cdot n \cdot \log_2 n$
  - Also $T(n) \leq A'' \cdot n \cdot \log_2 n$ für all $n \geq 2$, für eine Konstante A''

O-Notation — Motivation

- Primär interessiert uns oft
  - das "Wachstum" einer Funktion, z.B. einer Laufzeit $T(n)$
  - Die Werte der Konstanten (z.B. A") sind dabei oft sekundär
  - Und auch, wenn die Schranken erst ab $n \geq \ldots$ gelten
  - Zum Beispiel war beim Sortieren interessant, dass
    - die Laufzeit von MinSort "wächst wie" n^2
    - aber die Laufzeit von MergeSort "wächst wie" $n \cdot \log n$
  - Das wollen wir jetzt formaler machen, damit wir in Zukunft etwas schreiben bzw. sagen können wie:
    - Die Laufzeit des Algorithmus ist $O(n)$ "O von n"
    - Die Laufzeit des Algorithmus ist $\Omega(n)$ "Omega von n"
    - Die Laufzeit des Algorithmus ist $\Theta(n)$ "Theta von n"

O-Notation — Definition 1/5

- Vorweg
  - Wir betrachten Funktionen $f : N \rightarrow R$
    - N = die natürlichen Zahlen (oft: die Eingabegrößen)
    - R = die reellen Zahlen (oft: irgendwelche Kosten)
  - Zum Beispiel
    - $f(n) = 3 \cdot n$
    - $f(n) = 2 \cdot n \cdot \log n$
    - $f(n) = n^2 / 10$
    - $f(n) = n^2 + 3 \cdot n \cdot \log n - 4 \cdot n$

O-Notation — Definition 2/5

- Groß-O, Definition
  - Seien g und f zwei Funktionen $N \rightarrow R$
  - **Intuitiv:** Man sagt g ist Groß-O von f
    - wenn g "höchstens so stark wächst wie" f
    - es zählt die "Wachstumsrate", nicht die absoluten Werte!
  - **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 $f : N \rightarrow R$ ist
    - $O(f) = \{ g : N \rightarrow R \mid \exists n_0 \in N\ \exists C>0\ \forall n \geq n_0\ \ g(n) \leq C \cdot f(n) \}$
    - dabei heißt $\exists$ = "es existiert ..." und $\forall$ = "für alle ..."

O-Notation — Definition 3/5

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

also zu zeigen dass $\exists C > 0 \; \exists n_0 \in \mathbb{N} \; \forall n \geq n_0$

$5 \cdot n + 7 \leq C \cdot n$

für $\underbrace{n \geq 1}_{=: n_0}$

$5 \cdot n + 7 \leq 5 \cdot n + 7 \cdot n = \underbrace{12}_{=: C} \cdot n$

Alternativ:

für $n \geq 7$:

$5 \cdot n + 7 \leq 5 \cdot n + n = 6 \cdot n$ $\square$

O-Notation — Definition 4/5

- Groß-Omega, Definition + Beispiel
  - Analog wie Groß-O, nur mit "**mindestens** so stark wächst wie"
  - Definition: Für eine Funktion $f : N \to R$ ist

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

$$\forall n \geq \underbrace{1}_{=:n_0} \quad 5 \cdot n + 7 \geq \underbrace{5}_{=:C} \cdot n$$

O-Notation — Definition 5/5

- Groß-Theta
  - Intuitiv: analog mit "**gleich** stark wächst wie"
  - Definition: Für eine Funktion $f : N \to R$ ist

    $\Theta(f) = O(f) \cap \Omega(f)$

    $X \cap Y$: die Schnittmenge von X und Y
  - Zum Beispiel $5 \cdot n + 7 = \Theta(n)$
  - Beweis unter Verwendung der Definition von Θ :

$5 \cdot n + 7 \leq O(n)$
korrekt:
$5 \cdot n + 7 \in O(n)$
man schreibt:
$5 \cdot n + 7 = O(n)$
und nicht $\leq$

wir haben schon gesehen:

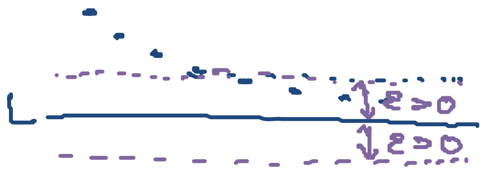
(1) $5 \cdot n + 7 = O(n)$

(2) $5 \cdot n + 7 = \Omega(n)$

(1) + (2) $\Longrightarrow$ $5 \cdot n + 7 = \Theta(n)$ ■

O-Notation — Grenzwerte 1/4

- In den bisherigen Beispielen ...
  - ... haben wir die Zugehörigkeit zu O(...) etc. gewissermaßen "zu Fuß" bewiesen, indem wir explizit das n_0 und das C bestimmt haben
  - Die Definitionen erinnern aber sehr 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 $n_0 \in N$ 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 $f : N \to R$ 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 1/3

- Beispiel für einen Beweis von einem Grenzwert

  (sollten Sie eigentlich in Mathe 1 schon mal gesehen haben)

$$\lim_{n \to \infty} 3 + \frac{1}{n} = 3$$

Beweis:

zu zeigen: für ein beliebiges $\varepsilon > 0$ gibt es ein $n_0 \in \mathbb{N}$ so dass $\forall n \geq n_0$

$$\underbrace{\left| \left(3 + \frac{1}{n}\right) - 3 \right|}_{1/n} \leq \varepsilon$$

für ein geg. $\varepsilon > 0$, wähle $n_0 = \lceil 1/\varepsilon \rceil$

Dann $n \geq n_0$: $\quad 1/n \leq \frac{1}{\lceil 1/\varepsilon \rceil} \leq 1/\tfrac{1}{\varepsilon} = \varepsilon$

O-Notation — Grenzwerte 2/3

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

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

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

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

O-Notation — Grenzwerte 3/3

- Beweis von (1)

zu zeigen: $f = O(g) \iff \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} < \infty$

„$\Longrightarrow$": $f = O(g)$

$\Longrightarrow$ (nach Def. von O) $\exists n_0, C\ \forall n \geq n_0\ f(n) \leq C \cdot g(n)$

$\Longrightarrow \forall n \geq n_0\ f(n)/g(n) \leq C$

$\Longrightarrow \lim\limits_{n\to\infty} f(n)/g(n) \leq C < \infty$

„$\Longleftarrow$": $\lim\limits_{n\to\infty} f(n)/g(n) = C < \infty$ für irgendein C

$\Longrightarrow$ (nach Def. lim)

$\exists n_0\ \forall n \geq n_0\ f(n)/g(n) \leq C + \underbrace{1}_{=:\varepsilon}$

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

$\Longrightarrow f = O(g)$

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, Kosten, etc. als $O(\ldots)$ oder $\Omega(\ldots)$ oder $\Theta(\ldots)$ ausdrückt, spricht man daher von

    asymptotischer Analyse
- Vorsicht
  - Asymptotische Analyse sagt nichts über das Laufzeit-verhalten bei "kleinen" Eingabegrößen ($n < n_0$) aus
  - Für $n < 2$ oder $n < 10$ ist das egal, da wird schon nichts Schlimmes passieren
  - Aber ...

O-Notation — Diskussion 2/2

- Beispiel
  - Algorithmus A hat Laufzeit $f(n) = 80 \cdot n$
  - Algorithmus B hat Laufzeit $g(n) = 2 \cdot n \cdot \log_2 n$
  - Dann ist $f = O(g)$ aber **nicht** $f = \Theta(g)$
  - Das heißt, A ist asymptotisch schneller als B
    - d.h. ab irgendeinem n_0 ist für alle $n \geq n_0$ $f(n) \leq g(n)$
  - Allerdings:

$$\forall n < 2^{40} = (2^{10})^4 \approx 1000^4 = 10^{12} = 1 \text{ Billionen}$$

$$g(n) = 2 \cdot n \cdot \underbrace{\log_2 n}_{< 40} = 80 \cdot n = f(n)$$

$\Rightarrow$ B schneller als A

Literatur / Links

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

    2.1 Asymptotic Notation
  - In Cormen/Leiserson/Rivest

    2.1 Asymptotic Notation
  - In Wikipedia

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

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