Grundzüge der Informatik 1

Vorlesung 3

Überblick

Überblick

- Diskussion der bisherigen Laufzeitanalyse
- Asymptotische Analyse
- O-Notation
- Ω-Notation

Wiederholung: Laufzeit InsertionSort

Laufzeitanalyse

- Laufzeit als Funktion der Eingabegröße
- Worst-Case Laufzeit $\boxed{T(n)}$ ist die maximale Laufzeit über alle Eingaben der Größe n (hier Anzahl der zu sortierenden Zahlen)
- Algorithmus InsertionSort hat eine Worst-Case Laufzeit von $T(n)= (3n^2+7n-8)/2$
- Beweisidee:
- Zeige, dass InsertionSort für jede Eingabe höchstens diese Laufzeit hat $\leq$
- Zeige, dass es für jedes n eine Eingabe gibt (hier: Absteigende Folge der Länge n), so dass der Algorithmus für diese Eingabe diese Laufzeit hat $\geq$

Diskussion

Laufzeitanalyse - Schwachpunkte

- Die konstanten Faktoren sind wenig aussagekräftig, da wir annehmen, dass jede Pseudocodeinstruktion einen Zeitschritt benötigt
- Gleichzeitig ist es aufwendig und fehlerträchtig, die konstanten Faktoren zu bestimmen

Erste Verbesserungsidee

- Die konstanten Faktoren ganz wegzulassen (als 1 anzunehmen)
- Man würde dann als Laufzeit statt $(3n^2+7n-8)/2$ einfach n^2+n-1 erhalten

Diskussion

Zweite Vereinfachungsidee

- Wenn wir nun die Funktion n^2+n-1 betrachten, dann gilt:
- n^2 wächst schneller als n und n wächst schneller als -1
- Wenn also n groß wird, so sind n und -1 zu vernachlässigen
- Daher ist es sinnvoll, sich nur auf den am schnellsten wachsenden Term zu beschränken
- Damit wäre das Laufzeitverhalten von InsertionSort durch $T(n) \approx n^2$ beschrieben

Diskussion

Zweite Vereinfachungsidee

- Wenn wir nun die Funktion n^2+n-1 betrachten, dann gilt:
- n^2 wächst schneller als n und n wächst schneller als -1
- Wenn also n groß wird, so sind n und -1 zu vernachlässigen
- Daher ist es sinnvoll, sich nur auf den am schnellsten wachsenden Term zu beschränken
- Damit wäre das Laufzeitverhalten von InsertionSort durch $T(n) \approx n^2$ beschrieben

- Wir werden nun eine Notation kennenlernen, die genau diese Ideen umsetzt

Universität zu Köln

Asymptotische Analyse

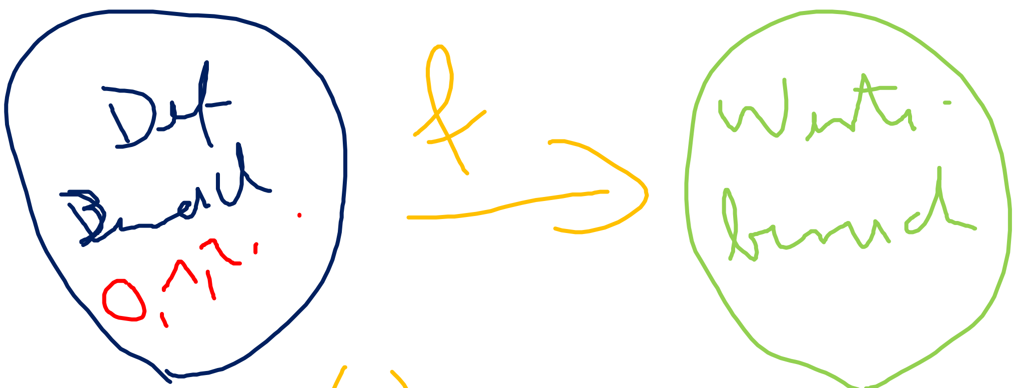
Grundideen (asymptotische Analyse)

- Ignoriere konstante Faktoren
- Betrachte das Verhältnis von Laufzeiten für $n \to \infty$
- Klassifizieren Laufzeiten durch Angabe von „einfachen Vergleichsfunktionen“

Landau Notation

Annahmen

- Wir werden typischerweise annehmen, dass unsere Funktionen als Definitionsbereich die natürlichen Zahlen $\mathbb{N}_0 = \{0,1,2,\ldots\}$ haben
- Dies ist normalerweise ausreichend, um Worst-Case Laufzeiten T(n) zu beschreiben



$f(2) = 11$

Landau Notation

Annahmen

- Wir werden typischerweise annehmen, dass unsere Funktionen als Definitionsbereich die natürlichen Zahlen $\mathbb{N}_0 = \{0,1,2,\ldots\}$ haben
- Dies ist normalerweise ausreichend, um Worst-Case Laufzeiten T(n) zu beschreiben

Wir werden die asymptotische Notation gelegentlich leicht missbräuchlich zu verwenden, z.B.

- indem der Definitionsbereich auf die reellen Zahlen erweitern wird,
- indem als Definitionsbereich eine Teilmenge der natürlichen Zahlen verwendet wird

Landau Notation

O-Notation

- $O(g(n)) = \{f(n)$: Es gibt positive Konstanten $\boxed{c}$ und $\boxed{n_0}$, so dass für alle $\boxed{n \geq n_0}$ gilt: $0 \leq f(n) \leq \boxed{c} \cdot g(n)\}$

Landau Notation

O-Notation

- $O(g(n)) = \{f(n)$: Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Interpretation

- $f(n) \in O(g(n))$ bedeutet, dass f(n) höchstens so stark wächst wie g(n) für $n \to \infty$
- Dabei ignorieren wir beim Wachstum konstante Faktoren
- Die O-Notation liefert eine obere Schranke
- Die Funktionen f(n) und g(n) müssen asymptotisch nicht-negativ sein (für n groß genug, sind die Funktionen nicht-negativ)

$T(n) \in O(n^2)$

Landau Notation

$O(g(n)) = \{f(n)$: Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $10n \in O(n)$

Beweis:

- Sei $f(n) = 10n$ und $g(n) = n$.
- Z.z.: Es gibt pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)$
- Wähle $c = 10$ und $n_0 = 1$
- Es gilt offensichtlich $0 \leq f(n) = 10n$ für alle $n \geq n_0 = 1$
- Es gilt außerdem $f(n) = 10n = c\ n = c\ g(n)$, d.h. insbesondere $f(n) \leq c \cdot g(n)$
- Damit ist $10n \in O(n)$

Universität zu Köln

Landau Notation

$O(g(n)) = \{f(n) :$ Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $10n \in O(n)$

Beweis:

- Sei $f(n) = 10n$ und $g(n) = n$.
- Z.z.: Es gibt pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)$
- Wähle $c = 10$ und $n_0 = 1$
- Es gilt offensichtlich $0 \leq f(n) = 10n$ für alle $n \geq n_0 = 1$
- Es gilt außerdem $f(n) = 10n = c\, n = c\, g(n)$, d.h. insb. $f(n) \leq c \cdot g(n)$ für $n \geq n_0 = 1$
- Damit ist $10n \in O(n)$

Landau Notation

$O(g(n)) = \{f(n) :$ Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $10n \in O(n^2)$

Beweis:

- Sei $f(n) = 10n$ und $g(n) = n^2$.
- Z.z.: Es gibt pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)$
- Wähle $c = 10$ und $n_0 = 1$
- Es gilt offensichtlich $0 \leq f(n) = 10n$ für alle $n \geq n_0 = 1$
- Es gilt außerdem $f(n) = 10n = c\,n \leq c\,n^2 = c\,g(n)$, d.h. insbesondere $f(n) \leq c \cdot g(n)$ für $n \geq n_0 = 1$
- Damit ist $10n \in O(n^2)$

Landau Notation

$O(g(n)) = \{f(n) :$ Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $n^2 \notin O(1000n)$

Beweis:

- Sei $f(n) = n^2$ und $g(n) = 1000n$.
- Z.z.: Für jede Wahl von positiven Konstanten c und n_0, gibt es ein $n \geq n_0$, so dass $0 \leq f(n) \leq c \cdot g(n)$ nicht erfüllt ist
- Seien also c und n_0 beliebige positive Konstanten
- Wähle $n = \max\{1001\, c, n_0\}$
- Dann ist $f(n) = n^2 \geq 1001 \cdot c \cdot n > c \cdot 1000\, n = g(n)$
- Damit gibt es keine Wahl von c und n_0, die die Definition erfüllt
- Somit gilt $n^2 \notin O(1000n)$

Universität zu Köln

Landau Notation

$O(g(n)) = \{f(n) :$ Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $O(1000n)=O(n)$

Beweis:

- Wir zeigen zunächst: $O(n) \subseteq O(1000n)$
- Sei $f(n) \in O(n)$ beliebig
- Dann gibt es pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot n$
- Damit gilt ebenfalls für alle $n \geq n_0$: $0 \leq f(n) \leq c \cdot 1000\, n$
- Somit ist $f(n) \in O(1000n)$
- Daraus folgt $O(n) \subseteq O(1000n)$

Landau Notation

$O(g(n)) = \{f(n)$: Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot g(n)\}$

Beispiel:

- $O(1000n)=O(n)$

Beweis:

- Als nächstes zeigen wir: $O(n) \supseteq O(1000n)$
- Sei $f(n) \in O(1000n)$ beliebig
- Dann gibt es pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt:
  $0 \leq f(n) \leq c \cdot 1000n$
- Damit gilt mit $c' = 1000c$ ebenfalls für alle $n \geq n_0$: $0 \leq f(n) \leq c' \cdot n$
- Somit ist $f(n) \in O(n)$
- Daraus folgt $O(n) \supseteq O(1000n)$
- Da $O(n) \subseteq O(1000n)$ und $O(n) \supseteq O(1000n)$ ist, folgt $O(1000n)=O(n)$

Landau Notation

Satz (Hierarchien bzgl. O-Notation)

- Es gilt für jede Konstante $b \geq 2$ und $\varepsilon > 0$,

1) $O(\log n) \subseteq O(\log^2 n) \subseteq O(\log^b n)$,
2) $O(\log^b n) \subseteq O(n^\varepsilon)$, und
3) $O(n^\varepsilon) \subseteq O(n) \subseteq O(n^2) \subseteq O(n^b)$.

$\log^2 n = (\log n)^2$

$O(1) \subseteq O(\log n)$

$O(n) \subseteq O(n \log n) \subseteq O(n^2)$

Universität zu Köln

Landau Notation

Beweis von (1) und (3)

- Zunächst zeigen wir $O(\log n) \subseteq O(\log^2 n)$:
- Sei $f(n) \in O(\log n)$
- Dann gibt es pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq f(n) \leq c \cdot \log n$
- Für $n \geq 2$ gilt $\log n \leq \log^2 n$
- Damit gilt für $n \geq \max\{2, n_0\}$: $0 \leq f(n) \leq c \cdot \log^2 n$
- Somit ist $f(n) \in O(\log^2 n)$ und damit folgt $O(\log n) \subseteq O(\log^2 n)$

- Analog zeigt man $O(\log^2 n) \subseteq O(\log^b n)$ und $O(n^\varepsilon) \subseteq O(n) \subseteq O(n^2) \subseteq O(n^b)$.

Landau Notation

$\log(n^x) = x \log n$

Beweis von (2)

- Sei $f(n) \in O(\log^b n)$
- Dann gibt es pos. Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt:
  $0 \leq f(n) \leq c \cdot \log^b n$
- Schreibe nun $n = m^{b/\varepsilon}$, d.h. $m = n^{\varepsilon/b}$
- Dann gilt $\log^b(n) = \log^b\left(m^{\frac{b}{\varepsilon}}\right) = \left(\frac{b}{\varepsilon}\right)^b \log^b(m) \leq \left(\frac{b}{\varepsilon}\right)^b m^b = \left(\frac{b}{\varepsilon}\right)^b n^{\varepsilon}$, da
- $0 \leq \log m \leq m$ für $m \geq 1$.
- Somit gilt für alle $n \geq n_0$: $f(n) \leq c \cdot \log^b n \leq c \left(\frac{b}{\varepsilon}\right)^b n^{\varepsilon}$
- Damit folgt $f(n) \in O(n^{\varepsilon})$

Landau Notation

„Kochrezept“ für die Verwendung der O-Notation

- Schritt 1: Entferne konstante Faktoren
- Schritt 2: Ist die resultierende Funktion eine Summe mehrerer Terme, so behalte nur einen Term mit dem größten Wachstum

Beispiel

$$\cancel{150}\, n^3 + \cancel{20}\, n \log n + \cancel{3}\, n^4 + \cancel{50}\, n\sqrt{n}$$

$$\boxed{n^4} + \cancel{n^3} + \cancel{n\sqrt{n}} + \cancel{n \log n}$$

$$n^{3/2}$$

$$O(n^4)$$

Universität zu Köln

Landau Notation

Satz

- Die Laufzeit von InsertionSort ist in $O(n^2)$.

Was bedeutet diese Aussage genau?

- Wenn n groß genug ist so gilt:
- Die Worst-Case Laufzeit des Algorithmus ist kleiner als eine Konstante mal n^2
- Die Worst-Case Laufzeit bei Eingabegröße n war aber die maximale Laufzeit über alle Eingaben dieser Größe
- Damit sagt der Satz, dass die Laufzeit jeder genügend großen Eingabe, höchstens eine Konstante mal n^2 ist

Landau Notation

Ω-Notation

- $\Omega(g(n)) = \{f(n)$: Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq c \cdot g(n) \leq f(n)\}$

Universität zu Köln

Landau Notation

Ω-Notation

- $\Omega(g(n)) = \{f(n)$: Es gibt positive Konstanten c und n_0, so dass für alle $n \geq n_0$ gilt: $0 \leq c \cdot g(n) \leq f(n)\}$

Interpretation

- $f(n) \in \Omega(g(n))$ bedeutet, dass f(n) mindestens so stark wächst wie g(n) für $n \to \infty$
- Dabei ignorieren wir beim Wachstum konstante Faktoren
- Die Ω -Notation liefert eine untere Schranke
- Die Funktionen f(n) und g(n) müssen asymptotisch nicht-negativ sein (für n groß genug, sind die Funktionen nicht-negativ)

Zusammenfassung

Schwachpunkte der bisherigen Laufzeitanalyse

- Konstante Faktoren wenig aussagekräftig
- Auswertung ist fehleranfällig
- Wollen vereinfachte Notation

O-Notation

- Lässt Konstanten und Terme niederer Ordnung weg
- Beschreibt Menge von Funktionen, die höchstens genau so schnell wachsen wie Vergleichsfunktion
- Kann genutzt werden, um obere Schranken für die Laufzeit anzugeben