Algorithmen und Datenstrukturen
Tutorium I

Michael R. Jung

20. - 23. 04. 2015

1. Organisatorisches
   - Kontakt
   - Übungsbetrieb
   - Klausuren
2. Landau-Notation
   - Definiton von $\mathcal{O}$
   - Logarithmen — Gesetze & Ableitung
   - Satz von l'Hôpital
3. Algorithmen
   - Analyse
   - Entwurf

E-Mail	jungmi@math.hu-berlin.de
Telefon	030 2093 **3146**
Büro	3.311
Sprechzeit	nach Vereinbarung
Webseiten	www.informatik.hu-berlin.de/∼mjung

www.informatik.hu-berlin.de/∼mjung/Tutorium/AlgoDat/tutorium.html

- Abgabe in Zweiergruppen innerhalb der gleichen Übungsgruppe
- eine Übungsserie erscheint alle zwei Wochen
- jede Serie muss bearbeitet werden
- in jede Serie sind 40 Punkte erreichbar
- insgesamt müssen 60% erreicht werden
- bei der Besprechung der Übungsserien in der Übung werden 2 bis 3 Gruppen ausgelost, die ihre Abgabe vorstellen werden
- sollte man das mehr als zweimal nicht schaffen, wird man den Übungsschein nicht erhalten

- Termine:
  - 11.08.2015
  - 15.09.2015
- keinerlei Hilfsmittel zugelassen
- Papier wird gestellt

Definition ($\mathcal{O}$)

Seien $f, g : \mathbb{N} \to \mathbb{N}$ zwei Funktionen. Dann ist $f \in \mathcal{O}(g)$ (auch $f = \mathcal{O}(g)$) genau dann, wenn

$$\exists c > 0 \, \exists n_0 \in \mathbb{N} \, \forall n > n_0 : f(n) \leq c \cdot g(n).$$

Darüber hinaus kann man diese Beziehung auch über Grenzwerte bestimmen, denn:

$$f \in \mathcal{O}(g) \Leftrightarrow \limsup_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

und sollte der Grenzwert existieren, dann auch einfach

$$f \in \mathcal{O}(g) \Leftrightarrow \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty.$$

Aufgabe 1

Zeigen Sie, dass $3n + 5 \in \mathcal{O}(n)$ gilt.

Lösung:

Für alle $n > 0$ (also $n \geq 1$) gilt: $3n + 5 \leq 3n + 5n = 8n$.

Oder: $\lim_{n\to\infty} \frac{3n+5}{n} = \lim_{n\to\infty} 3 + \frac{5}{n} = 3 < \infty$.

Achtung! $\log = \log_2$

In der theoretischen Informatik meint man mit $\boxed{\log n}$ fast immer $\boxed{\log_2 n}$!

Es gilt:

- $\log a^b = b \log a$
- $\log(a \cdot b) = \log a + \log b$
- $\log(\frac{a}{b}) = \log(a \cdot b^{-1}) = \log a + \log b^{-1} = \log a - \log b$
- $2^{\log a} = a = e^{\ln a}$

Daraus lässt sich nun auch die Ableitung für $\log n$ herleiten:
$2^{\log n} = e^{\ln n} \Leftrightarrow \ln 2^{\log n} = \ln e^{\ln n} \Leftrightarrow \log n \cdot \ln 2 = \ln n \cdot \ln e \Leftrightarrow \log n = \frac{\ln n}{\ln 2}$
und somit $(\log n)' = (\frac{\ln n}{\ln 2})' = \frac{1}{|n| \ln 2}$, also für natürliche Zahlen $n : \frac{1}{n \ln 2}$.

Satz (l'Hôpital für $n \to \infty$)

Falls $\lim_{n\to\infty} f(n) = \lim_{n\to\infty} g(n) = 0$ *(oder* $\lim_{n\to\infty} f(n) = \lim_{n\to\infty} g(n) = \infty$*) und* $\lim_{n\to\infty} \frac{f'(n)}{g'(n)}$ *existiert, so gilt:*

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}.$$

Aufgabe 2

Zeigen Sie, dass $\log n \in \mathcal{O}(\sqrt{n})$ und $n \log n \in \mathcal{O}(2^n)$ gilt.

Lösung:

$$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} \overset{\text{l'Hôp}}{=} \lim_{n\to\infty} \frac{\frac{1}{n \ln 2}}{\frac{1}{2} n^{-\frac{1}{2}}} = \lim_{n\to\infty} \frac{2\sqrt{n}}{n \ln 2} = \lim_{n\to\infty} \frac{2}{\cancel{\sqrt{n}}}\frac{\cancel{\sqrt{n}}}{\cancel{n}\sqrt{n}} \frac{1}{\ln 2} = 0.$$

$$\lim_{n\to\infty} \frac{n \log n}{2^n} \overset{\text{l'Hôp}}{=} \lim_{n\to\infty} \frac{1 \log n + \frac{\cancel{n}}{\cancel{n} \ln 2}}{2^n \ln 2} \overset{\text{l'Hôp}}{=} \lim_{n\to\infty} \frac{\frac{1}{n \ln 2}}{2^n (\ln 2)^2} = \lim_{n\to\infty} \frac{1}{n 2^n (\ln 2)^3} = 0.$$

Aufgabe 3

Analysieren Sie den folgenden Algorithmus hinsichtlich des berechneten Ergebnisses und der benötigten Laufzeit!

Algorithmus Alg

```
Input: Array A der Länge n über Zahlen>0
B[0]=A[0],B[1]=A[0],B[2]=A[0]
for i=n-1 downto 0
  for j=0 to n-1
    for k=n-1 downto 0
      x=A[i]*A[j]/A[k]
      y=B[0]*B[1]/B[2]
      if x<y then
        B[0]=A[i]
        B[1]=A[j]
        B[2]=A[k]
      endif
    endfor
  endfor
endfor
Output: B
```

- Ergebnis:
  Alg durchläuft alle möglichen Tripel (x, y, z) aus `A`. Dabei berechnet er $\frac{xy}{z}$ und überprüft, ob das Ergebnis kleiner ist, als das Ergebnis des Tripels in `B`. Falls dem so ist, wird das Tripel in `B` durch das aktuell überprüfte ersetzt. Folglich sucht er das Tripel, welches $\frac{xy}{z}$ minimiert.
- Laufzeitanalyse:
  Zur Vereinfachung nehmen wir an, dass Multiplikation und Division Aufwand 1 haben.

```
Input: Array A der Länge n über Zahlen>0
Array B der Länge 3
B[0]=A[0],B[1]=A[0],B[2]=A[0]    3
for i=n-1 downto 0               n
  for j=0 to n-1                 n
    for k=n-1 downto 0           n
      x=A[i]*A[j]/A[k]           3
      y=B[0]*B[1]/B[2]           3
      if x<y then                1
        B[0]=A[i]                1
        B[1]=A[j]                1
        B[2]=A[k]                1
      endif
    endfor
  endfor
endfor
Output:       Summe(worst-case): 10n^3 + 3
```

Aufgabe 4

Geben Sie einen Linearzeitalgorithmus an, der die gleiche Funktion berechnet!

Algorithmus Alg'

```
Input: Array A der Länge n über Zahlen>0
min=A[0],max=A[0]                       2
for i=1 to n-1                          n-1
  if A[i]< min then min=A[i] endif      2
  if A[i]> max then max=A[i] endif      2
endfor
Output: (min,min,max)           Summe:  4n-2
```

Bei genauer Analyse zeigt sich, dass die Worst-Case-Laufzeit von Alg' sogar nur 3n-1 beträgt, da keinesfalls `min` und `max` gleichzeitig geändert werden. Diese Laufzeit wird bei einem sortierten Array auch erreicht.