Algorithmen und Datenstrukturen 1

ALGO1 · SoSe-2021 · tcs.uni-frankfurt.de/algo1

Wochenplan: Analyse von Algorithmen

Revision 0e24027 (2021-06-28)

Vorbereitung

Lies CLRS Kapitel 3 sowie 4.3–4.5 und schau das Video der Woche.

Dienstag

Aufgabe 1 (Asymptotisches Wachstum, einfach). Ordne die folgenden Funktionen in aufsteigender Reihenfolge nach ihrem asymptotischem Wachstum. Das heißt, $f(n)$ kommt vor $g(n)$, wenn $f(n) = o(g(n))$ gilt.

$$n \log n \qquad n^2 \qquad 2^n \qquad n^3 \qquad \sqrt{n} \qquad n$$

Aufgabe 2 (Θ-Notation). Vereinfache die folgenden Funktionen in Θ-Notation.

$$\begin{array}{cc}
n^2 + n^3/2 & 8\log_2^7 n + 34\log_2 n + \frac{1}{1000}n \\
2^n + n^4 & 2^n \cdot 7 + 5\log_2^3 n \\
\log_2 n + n\sqrt{n} & n(n^2 - 18)\log_2 n \\
n(n-6) & n\log_2^4 n + n^2 \\
4\sqrt{n} & n^3\log_2 n + \sqrt{n}\log_2 n \\
99\sin^2 n + 99\cos^2 n & \frac{100n}{\log n} + \frac{1}{n}
\end{array}$$

Aufgabe 3 (Looping Louie). Analysiere die Laufzeit für die folgenden Funktionen in n und gib das Ergebnis in O-Notation an.

```
Loop1(n)
i = 1
while i <= n do
  print "*"
  i = 2*i
return
```

```
Loop2(n)
i = 1
while i <= n do
  print "*"
  i = 5*i
return
```

```
Loop3(n)
for i = 1 to n do
  j = 1
  while j <= n do
    print "*"
    j = j*2
return
```

Aufgabe 4 (Asymptotische Aussagen). Sind die folgenden Aussagen wahr oder falsch?

$$\begin{array}{cc}
\frac{1}{20}n^2 + 100n^3 = O(n^2) & \frac{n^3}{1000} + n + 100 = \Omega(n^2) \\
\log_2 n + n = O(n) & 2^n + n^2 = \omega(n) \\
2^{\log_2 n} = O(n) & \log_4 n + \log_{16} n = \Theta(\log n) \\
n^3(n-1)/5 = \Theta(n^3) & n^{1/4} + n^2 = \Theta(n) \\
\log_2^2 n + n = \Theta(n) & 2^{\log_4 n} = \Theta(\sqrt{n}) \\
n^{1.9}\log^9 n = o(n^2/\log n) & n \cdot (n \bmod 7) = \Theta(n) \\
f(n) = \omega(g(n)) \Longrightarrow f(n) = \Omega(g(n)) & f(n) = O(g(n)) \Longrightarrow f(n) = o(g(n))
\end{array}$$

Aufgabe 5 (Master der Asymptotik). Löse die folgenden Rekursionsgleichungen in Θ-Notation als Funktion von n. Benutze Rekursionsbäume um die Gleichungen für A, B, C zu lösen, und für D, E, F benutze das Mastertheorem.

$$\begin{array}{ccc} A(n) = 2A(n/4) + \sqrt{n} & B(n) = 2B(n/4) + n & C(n) = 2C(n/4) + n^2 \\ D(n) = 8D(n/2) + 487n^2 & E(n) = 2E(n/4) + n^{0.4} & F(n) = 9F(n/3) + n^2 \end{array}$$

Donnerstag

Aufgabe 6 (Verdopplungen). Löse die folgenden Teilaufgaben.

a) (einfach) Algorithmus A benötigt genau $7n^3$ Operationen für eine Eingabe der Länge n. Wie viele Operationen mehr benötigt A bei doppelter Eingabelänge?

b) Betrachte die Laufzeiten für einen Algorithmus B:

Eingabelänge (Bits)	1000	2000	3000	4000	5000
Dauer [Sekunden]	5	20	45	80	125

Schätze die Laufzeit von B auf einer 6000 Bit langen Eingabe. Was ist vermutlich die asymptotische Laufzeit von B? Drück deine Vermutung in Abhängigkeit von Eingabelänge n in O-Notation aus.

c) Algorithmus C benötigt für jede Verdoppelung der Eingabelänge 3 Sekunden länger. Gib die asymptotischen Laufzeit von C in Abhängigkeit von Eingabelänge n in O-Notation an.

Aufgabe 7 (Asymptotische Eigenschaften). Löse die folgenden Teilaufgaben.

a) Seien $f(n)$ und $g(n)$ asymptotisch nicht negative Funktionen. Beweise, dass folgendes gilt: $\max(f(n), g(n)) = \Theta(f(n) + g(n))$

b) Erkläre warum die Aussage „Die Laufzeit von Algorithmus D ist mindestens $O(n^2)$“ keinen Sinn ergibt.

c) Gilt $2^{n+1} = O(2^n)$? Gilt $2^{2n} = O(2^n)$? Beweise deine Behauptung.

d) Beweise, dass $\log_2(n!) = O(n \log n)$ gilt.

e) (schwer) Beweise, dass $\log_2(n!) = \Omega(n \log n)$ gilt. Beweise, dass $\log_2(n!) = \Theta(n \log n)$ gilt.

Aufgabe 8 (Maximale Teilfelder). Sei $A \in \mathbb{Z}^n$ als ein Feld $A[0, \ldots, n-1]$ gespeichert. Ein Teilfeld von A ist ein genau dann ein *maximales Teilfeld* $A[i..j]$ mit $0 \leq i \leq j \leq n-1$, wenn die Summe $A[i] + A[i+1] + \cdots + A[j]$ maximal über alle möglichen Teilfelder ist. Löse die folgenden Aufgaben.

a) (einfach) Schreib einen Algorithmus, der ein maximales Teilfeld von A in Laufzeit $O(n^3)$ findet.

b) Schreib einen Algorithmus, der ein maximales Teilfeld von A in Laufzeit $O(n^2)$ findet.

c) (schwer) Schreib einen *Divide and Conquer* Algorithmus, der ein maximales Teilfeld von A in Laufzeit $O(n \log n)$ findet.

d) (sehr schwer) Schreib einen Algorithmus, der ein maximales Teilfeld von A in Laufzeit $O(n)$ findet.