Max Wisniewski , Alexander Steen

Tutor: Tilmann

Aufgabe 1

Die Funktionen der Aufgabe sollen derart geordnet werden, so dass $g_i \in \Omega(g_{i+1})$ gilt. Geben Sie auch an, wenn sogar $g_i = \Theta(g_{i+1})$ ist.

Die Folge erfüllt die Eigenschaft und enthält die Elemente, die geordent werden müssen:

$(g_i)_{1\leq i\leq 10} = (n^{\frac{1}{\log n}}, \ln n, \log^2 n, (\sqrt{2})^{\log n}, n^2, 4^{logn}, (\lceil \log n \rceil)!, n^{log(log(n))}, 2^n, 2^{(2^n)})$

Beweis

1. $2^n \in \Omega(2^{(2^n)})$:
   $\lim\limits_{n\to\infty} \frac{2^n}{2^{(2^n)}} = \lim\limits_{n\to\infty} \frac{2^n \cdot 1}{2^n \cdot 2^{(2^n)-n}} = \lim\limits_{n\to\infty} \frac{1}{2^{(2^n)-n}}$
   Da 2^n stärker wächst als n, gilt:
   $\lim\limits_{n\to\infty} \frac{1}{2^{(2^n)-n}} = 0$
   Damit gilt nach konvergenz Kriterium $2^n \in \Omega(2^{(2^n)}) \Rightarrow g_9 \in \Omega(g_{10})$
2. $n^{\log(\log(n))} \in \Omega(2^n)$:
   Es gilt: $n^{\log\log n} = 2^{\log n \cdot \log\log n} = 2^{\log(n+\log n)}$

   $$\lim_{n\to\infty} \frac{2^{\log(n+\log n)}}{2^n} = \lim_{n\to\infty} 2^{n-\log(n+\log n)} = 0 \Leftarrow \lim_{n\to\infty} n - \log(n+\log n) = \infty$$

   Dies ist der Fall, wenn n stärker wächst als $\log(n + \log n)$.
3. $(\lceil \log n \rceil)! \in \Omega(n^{log(\log(n))})$:
   Später
4. $4^{\log(n)} \in \Omega((\lceil \log n \rceil)!)$:
   Wie gehabt
5. $n^2 \in \Theta(4^{\log(n)})$:
   Wir zeigen an dieser Stelle, das gilt: $n^2 = 4^{\log(n)}$. Damit gilt die Beziehung für Θ sofort.
   $4^{\log n} = (2^2)^{\log n} = 2^{2\cdot\log n} = 2^{\log n^2} = n^2$
   $\Rightarrow g_5 \in \Theta(g_6)$
6. $(\sqrt{2})^{\log n} \in \Theta(n^2)$:
   Ersteinmal gilt : $(\sqrt{2})^{\log n} = (2^{\frac{1}{2}})^{\log n} = 2^{\frac{1}{2}\cdot\log n} = 2^{\log\sqrt{n}} = \sqrt{2}$
   Nun wenden wir wieder das Konvergenzkreterium an:
   $\lim\limits_{n\to\infty} \frac{\sqrt{2}}{n^2} = \lim\limits_{n\to\infty} \frac{1}{n^{1.5}} = 0$
   $\Rightarrow (\sqrt{2})^{\log n} \in \Theta(n^2) \Rightarrow g_4 \in \Omega(g_5)$
7. $\log^2 n \in \Omega(\sqrt{n})$:

   $$\lim_{n\to\infty} \frac{\log^2 n}{\sqrt{n}} = \lim_{n\to\infty} \frac{\log^2 n}{\log(2^{\sqrt{n}})} = 0$$

   Sagt Wolframalpha

8. $\ln n \in \Omega(\log^2 n)$:

$$\lim_{n\to\infty} \frac{\ln n}{\log^2 n} = \lim_{n\to\infty} \frac{(\ln 2)(\log n)}{\log^2 n} = \lim_{n\to\infty} \frac{\ln 2}{\log n} = 0$$

Nach Konvergenzkriterium gilt: $g_2 \in \Omega(g_3)$

9. $n^{\frac{1}{\log n}} \in \Omega(\ln n)$:
Zunächst gilt: $n^{\frac{1}{\log n}} = 2^{\frac{\log n}{\log n}} = 2$.
Daraus folgt offensichtlich:
$\lim_{n\to\infty} \frac{2}{\ln n} = 0 \Rightarrow g_1 \in \Omega(g_2)$

Aufgabe 2

Sei n die Anzahl der verschiedenen Sammelbilder. Sei X Zufallsvariable für die Anzahl der benötigten Packungen Müsli, bis wir alle n Sammelbilder haben. Die Wahrscheinlichkeit für ein bestimmtes Bild ist gleichverteilt.

(a)

tbd

(b)

tbd

(c)

tbd

Aufgabe 3

(a)

Selectionsort

Beschreibung: Solange bis die zu sortierende Liste leer ist, sucht Selectionsort das kleinste Element, löscht dieses aus der Liste und fügt es hinten an eine am Anfang neu erzeugte Liste an.

Laufzeit: Um das kleinste Element in einer unsortierten Liste zu finden, muss man sich jedes Element einmal ansehen. Bei einer Liste der Länge n bedeutet dies $T_{smallest}(n) = n$.
Nach der obigen Beschreibung nehmen wir n mal das kleinste Element aus der Liste, wobei die Liste in jedem Schritt um ein Element schrumpft. Um das Element hinten an die neue Liste zu hängen, können wir konstante laufzeit annehmen. (LinkedList rear oder Array auf das ende + 1).

$$\begin{aligned} \Rightarrow T(n) &= \sum_{i=0}^{n-1} i \\ &= \frac{n\cdot(n-1)}{2} \\ &= \frac{1}{2}\left(n^2 - n\right) \end{aligned}$$

Mergesort

Beschreibung: Teile die Liste rekursiv in 2 gleichgroße Listen, bis die Teile trivial zu lösen sind, z.B. bei einem Element, und vereinige danach die sortierten Teillisten auf dem Weg den Rekursionsbaum hoch wieder, in der richtigen Reihenfolge.

Laufzeit: Wir teilen die Liste in 2 gleichgroße Teile (da n Zweierpotenz ist, geht diese Partitionierung immer genau gleich groß werden). Diese werden wieder bearbeitet. Nach diesem Schritt werden wir die Listen mergen. Das mergen funktioniert so, dass ich nach einem Vergleich zwischen den jeweils kleinten Elementen beider Listen (die ganz vorne stehen) eins in die neue Liste einfügen kann. Das führt dazu, dass ich n Vergleiche brauche. Dies ist die Worstcase Laufzeit, wenn immerabwechselnd ein Element aus den Listen genommen wird.
Betrachten wir den Berechnungsgraphen sehen wir, dass der Baum nach $logn$ seine Blätter hat. Dies gilt, da wir nach jedem Schritt die Listen halbieren und erst aufhören, wenn die Listen die Länge 1 haben.

$$\begin{aligned} \Rightarrow T(n) &= 2 \cdot T(\tfrac{n}{2}) + n \\ &= 4 \cdot T(\tfrac{n}{4}) + 2n + n \\ &= 8 \cdot T(\tfrac{n}{8}) + 4n + 2n + n \\ &\overset{Tiefe}{=} n \cdot \log n \end{aligned}$$

(b)

tbd

(c)

tbd