TH Brandenburg
Online Studiengang Medieninformatik
Fachbereich Informatik
Algorithmen und Datenstrukturen

Einsendeaufgabe 1

Sommersemester 2021
Abgabetermin 18.04.2021

Maximilian Schulke
Matrikel-Nr. 20215853

1 Zweitkleinstes Element einer Folge

Das zweitkleinste Element einer Folge von n ≥ 2 Zahlen soll bestimmt werden.

1.1 Algorithmus in Pseudocode

```
def second_minimum(list):
    second = list[0]
    minimum = list[0]

    for n in list[1:]:
        if n > minimum:
            second = n
            break

    for n in list[1:]:
        if n < minimum:
            second = minimum
            minimum = n

    return second
```

1.2 Laufzeit-Analyse

Der Algorithmus braucht im **Best-Case n** Vergleiche, liegt also dementsprechend in $\Omega(n)$. Der Best-Case tritt ein, wenn direkt das zweite Element größer als das erste ist, da dann die erste Schleife nach dem ersten Schritt abgebrochen wird und die 2. Schleife immer genau $n-1$ vergleiche ausführt.

Er braucht im **Worst-Case 2(n - 1)** Vergleiche und liegt daher in $O(n)$. Der Worst-Case kommt zustande wenn wir z.B. eine List der Länge n betrachten, die n mal das gleiche Element enthält. Dann benötigen wir beim der ersten und der zweiten Schleife $n - 1$ Vergleiche.

2 Asymptotische Notation

Gegeben sei die Funktion $f(n) = 2n^2 + 3n \log_2 n - 72$

2.1 Beweis von $f(n) \in O(n^2)$

$$\begin{aligned} f(n) &= 2n^2 + 3n \log_2 n - 72 \\ &\leq 2n^2 + 3n \log_2 n \\ &\leq 2n^2 + 3n^2 \\ &= 5n^2 \end{aligned}$$

Somit können wir sagen, dass mit $c \geq 5$ und $n_0 = 1$ die Behauptung $f(n) \in O(n^2)$ gilt

2.2 Beweis von $f(n) \in \Omega(n^2)$

$$\begin{aligned} f(n) &= 2n^2 + 3n \log_2 n - 72 \\ &\geq 2n^2 - 72 \\ &\geq n^2 \end{aligned}$$

Nun können wir n_0 als Schnittpunkt der beiden Funktionen $2n^2 - 72$ und n^2 berechnen.

$$\begin{aligned} 2n^2 - 72 &= n^2 \mid -2n^2 \\ -72 &= -n^2 \mid * -1 \\ 72 &= n^2 \\ n &= \sqrt{72} \end{aligned}$$

Also, mit $c = 1$ und $n_0 = \lceil \sqrt{72} \rceil = 9$ gilt $f(n) \in \Omega(n^2)$

2.3 Gilt $f(n) \in \Theta(n^2)$?

$\Theta(g)$ ist im Skript mit der *Definition 2.5* als $\{ f \mid f \in O(g) \wedge f \in \Omega(g) \}$ definiert.

Somit wissen wir, dass $f(n) \in \Theta(n^2)$, da wir in 2.1 und 2.2 gezeigt haben, dass $f \in O(g)$ und $f \in \Omega(g)$ gelten.

3 Average-Case-Aufwand der binären Suche

3.1 Durchschnittliche Anzahl der Vergleiche für einen Hit

Element	0	1	2	3	4	5	6	7	8	9
Vergleiche	3	2	3	4	1	3	4	2	3	4

Macht in Summe 29 Vergleiche und somit $\frac{29}{10} = 2.9$ Vergleiche im Durchschnitt.

3.2 Summenformel für Vergleiche bei $2^k - 1$ Elementen

Beispiel für $k = 3$

			3			
	1				5	
0		2		4		6

Beispiel für $k = 4$

							7							
			3								11			
	1				5				9				13	
0		2		4		6		8		10		12		14

Es gibt bei $2^k - 1$ immer einen perfekten, gleichmäßigen Baum und immer genau $\log_2 n$ bzw. k Ebenen. Auf (einer 0 indizierten) Ebene i haben wir den Baum i Mal geteilt und vergleichen $(i+1) * 2^i$ Elemente. Wenn wir nun alle Ebenen addieren möchten, um die gesamt Anzahl der verglichenen Elemente bekommen möchten, müssen wir lediglich alle Ebenen addieren. Also bei $k = 4$ wären wir bei $1 * 2^0 + 2 * 2^1 + 3 * 2^2 + 4 * 2^3$ Vergleichen. Dies lässt sich durch die gaußsche Summenformel eleganter (und allgemeingültiger) Zusammenfassen zu $\sum_{i=0}^{k-1} (i+1) * 2^i$.