Nicole Kaiser fl Tutorium: Do. 12-14 Uhr fl Tutor: Martin Klossek

05.12.2001

Theoretische Informatik 1, WS 2001/2002

Übungsblatt 7

Keine Garantie für die Korrektheit der hier vorgestellten Lösungen.

Aufgabe 7.1

(a)

Vergleichsorientiert Sortierverfahren verhalten sich genauso wie Vergleichsbäume. Nach Kapitel 3.4 kann die höchste Anzahl von Vergleichen über die Tiefe des Sortierbaums bestimmt werden.

Damit ist z.z.: Tiefe des Sortierbaums ist $\Omega(n\cdot\log(d))$.

Der Abstand eines Elements von seiner endgültigen Position ist d. Um ein Element an seine endgültige Position zu bringen, werden höchstens $2(d-1)+1=2d-1$ Vergleiche pro Element benötigt. Insgesamt also $n(2d-1)$ Vergleiche für alle n Elemente. Die Tiefe des Sortierbaums ist also $n(2d-1)$.

Behauptung:

$$n(2d-1)=\Omega(n\cdot\log(d))$$

Beweis:

Nach Definition 1.10 gilt $f=\Omega(g) \Leftrightarrow g=O(f) \Leftrightarrow g\le c\cdot f$.

$n\cdot\log d\le n(2d-1)$, weil $n\le c'n$ und $\log d \le d\le 2d-1$ nach Lemma 1.17

Damit ist die Worst-Case Laufzeit $\Omega(n\cdot\log(d))$.

Aufgabe 7.2

(a)

Sei A die Liste der $n-\frac{n}{\log n}$ vorsortierten Zahlen und B die Liste der $\frac{n}{\log n}$ unsortierten Zahlen.

Verfahren:

I. Sortiere B mit Mergesort. Die Laufzeit dafür ist $\Theta\left(\frac{n}{\log n}\log\left(\frac{n}{\log n}\right)\right)$.

II. Sortiere die nun sortiere Liste B in A ein. Sei von b_1 bis $b_{n/\log n}$ die Liste aufsteigend sortiert. Vergleiche nun b_1 mit allen a_i bis die passende Position für b_1 gefunden ist und füge es in A ein. Vergleiche weiter b_2 mit allen a_i nach der Position von b_1 bis die passende Stelle für b_2 gefunden ist und füge es ein. Analog weiter mit allen weiteren b_i .

Laufzeit:

I. $\frac{n}{\log n}\log\left(\frac{n}{\log n}\right)=\frac{n}{\log n}(\log n-\log(\log n))=n-\frac{n\log(\log n)}{\log n}$ da nach Lemma 1.17 $\log n<n$

$n-\frac{n\log(\log n)}{\log n}<n \quad \Rightarrow O(n)$

II. a.) Insgesamt wird A nur einmal durchlaufen, um alle b_i einzufügen. $\Rightarrow O(n)$

b.) Die passenden Position für ein b_i ist erst gefunden, wenn ein a_i erreicht wurde, dass größer als

das b_i ist. b_{i+1} wird dann auch mit diesem a_i verglichen. Dafür zusätzliche $\frac{n}{\log n}-1$ Vergleiche. Da aber $\frac{n}{\log n}<n \quad \Rightarrow O(n)$.

Gesamte Laufzeit: $O(n)$.

(b)

Die Daten können in lineare Laufzeit nach Bundesländern sortiert werden. Verwende dazu Distribution Counting. Für jedes Bundesland wird ein Korb bereitgestellt.

$$Anzahl\ Körbe = m = 16$$

Es kann davon ausgegangen werden, dass es mehr Verkehrssünder als Bundesländer gibt:

$$n > m$$

Damit sortiert Distribution Counting in $O(n)$.