

Max Wisniewski , Alexander Steen

Tutor: Lena Schlipf

Aufgabe 1

Sei $S = \{s_1, s_2, \dots, s_n\}$ eine Menge von n paarweise verschiedenen Elementen aus einem totalgeordneten Universum. Seien w_1, w_2, \dots, w_n positive Gewichte, so dass das Element s_i Gewicht w_i hat. Es gelte $\sum_{i=1}^n w_i = 1$. Gesucht ist der *gewichtete Median* von S .

- (a) Angenommen, Sie haben eine Funktion, welche den gewichteten Median in Zeit $T(n)$ bestimmt. Zeigen Sie, wie man den (normalen) Median in Zeit $O(n) + T(n)$ berechnen kann.

Beweis: Als Eingabe für unseren Algorithmus (Median) bekommen wir eine Liste von Werten (s_1, \dots, s_n) . Diese erweitern wir nun auf die Form für den gewichteten Median. Dazu legen wir eine 2. Liste mit den Werten an (oder erweitern die andere Liste auf Tupel (s_i, w_i) , was beides auf das selbe hinaus läuft). Für die Gewichte wählen wir $\forall 1 \leq i \leq n : w_i = \frac{1}{n}$. Auf die neue Struktur können wir den Algorithmus für den *gewichteten Median* anwenden.

Laufzeit: Wir konstruieren zunächst die neue Liste. Dafür gehen wir einmal drüber und erzeugen Tupel. Das Gewicht können wir in $O(1)$ erstellen und den Tupel auch. Dies tun wir für n Elemente. Je nachdem, ob wir n besitzen oder noch suchen müssen, haben wir als Laufzeit n oder $2n$. Wir benötigen aber für das Erstellen der neuen Eingabe $O(n)$ Schritte. Danach führen wir den gegebenen Algorithmus aus.
 $\Rightarrow T_{\text{Median}}(n) = T_{\text{gewichteter Median}}(n) + O(n)$

Median: Bleibt zu zeigen, dass das Ergebnis s_g der Median ist:
 Kleinere Elemente, sei n_k die Anzahl der Elemente, die kleiner als s_g sind:

$$\sum_{s_i < s_g} w_i \stackrel{\text{Gewichte}}{=} \sum_{s_i < s_g} \frac{1}{n} \stackrel{\text{Def.: } n_k}{=} \frac{n_k}{n} \stackrel{\text{Def.}}{\leq} \frac{1}{2}$$

$$\Leftrightarrow n_k \leq \frac{n}{2}$$

Kleinere Elemente, sei n_g die Anzahl der Elemente, die größer sind als s_g sind:

$$\sum_{s_i > s_g} w_i \stackrel{\text{Gewichte}}{=} \sum_{s_i > s_g} \frac{1}{n} \stackrel{\text{Def.: } n_g}{=} \frac{n_g}{n} \stackrel{\text{Def.}}{<} \frac{1}{2}$$

$$\Leftrightarrow n_g < \frac{n}{2}$$

Da gelten muss, dass $n_g + n_k + 1 = n$ erfüllen unsere beiden Ergebnisse die Bedingungen für den normalen Median.

- (b) Zeigen Sie, wie man den gewichteten Median in $O(n \cdot \log n)$ Zeit berechnen kann.

Beweis: Wir wählen uns einen geeigneten Sortieralgorithmus. Hier bietet sich Mergesort an. Aus dem letzten Übungszettel wissen wir, dass $T_{\text{Mergesort}}(n) = n \cdot \log n - (n - 1)$ ist.

Nachdem wir unsere Liste sortiert haben tun wir folgendes:

Schritt 1 Erzeuge $sum = 0$ Laufsummenvariable mit 0 und indexvariable $i = 0$

Schritt 2 Rechne $sum \leftarrow sum + w_i$.

Schritt 3 Ist $sum \leq \frac{1}{2}$ inkrementiere i um eins und mache mit Schritt 2 weiter.

Schritt 4 (Ist dies nicht der Fall) Gebe s_i zurück.

Was wir als erstes sehen, ist dass die Summe der Gewichte, die kleiner als der gewichtete Median sind, $\leq \frac{1}{2}$ da wir beim ersten mal, wenn es größer ist aus der Schleife gehen und dort nur den Median drauf gerechnet haben. Da wir beim finden über $\frac{1}{2}$ gekommen sind, wissen wir, dass aus der Bedingung, dass die Gesamtgewichte = 1 sind, dass die Gewichte der Elemente, die größer sind $\geq \frac{1}{2}$ sein müssen.

Laufzeit: Die erste Überlegung ist, ob die Schleife abbricht.

Eine einfache Antwort wäre, dass wir alle Gewichte aufsummieren, d.h. wir wissen im n -ten Schritt, dass $sum = 1$ ist. Die Abbruchbedingung wird also spätestens nach n Schritten erfüllt. Da wir aber äußerste Beweisfetischisten sind, werden wir für die Terminierung noch einen formellen Beweis am Ende des Übungsblattes geben.

Für die Laufzeit ergibt sich nun: $T(n) = T_{\text{Mergesort}} + n \cdot (c) + d$, wobei c die konstanten Kosten sind, für das aufsummieren der Gewichte und inkrementieren des indexes sind und d die Fixkosten für das anlegen der beiden Variablen sind.

$\Rightarrow T(n) = n \cdot \log n - (n - 1) + n \cdot c + d \in O(n \log n)$.

- (c) Zeigen Sie, wie man den gewichteten Median in $O(n)$ Zeit finden kann, wenn ein Linearzeitalgorithmus zum Finden des normalen Medians zur Verfügung steht.

Aufgabe 2

In der Vorlesung wurden zur Berechnung des BFPRT-Algorithmus die Menge in 5er Gruppen unterteilt. Untersuchen Sie, wie sich der Algorithmus für $k = 2n + 1$, $n \in \mathbb{N}$ verhält.

Aufgabe 3

- (a) Für zwei ganzzahlige Vektoren $x = (x_1; x_2; \dots; x_n)$ und $y = (y_1; y_2; \dots; y_n)$ mit $0 \leq x_i, y_i \leq M$ und einen Wert $u > M$ betrachten wir die Zahlen

$$a = x_1 u^n + x_2 u^{n-1} + \dots + x_n u^1$$

und

$$b = y_1 u^n + y_2 u^{n-1} + \dots + y_n u^1.$$

Zeigen Sie: $a = b \Leftrightarrow x = y$, $a < b \Leftrightarrow x < y$ (lexikographisch).

- (b) Entwerfen Sie einen Algorithmus, der für zwei gegebene Folgen nichtnegativer Zahlen $x = (x_1; \dots; x_m)$ und $y = (y_1; \dots; y_n)$ in $\Theta(n \log m)$ entscheidet, ob x als Teilfolge $(y_{i+1}, \dots, y_{i+m})$ in y vorkommt ($0 \leq i \leq n - m$). Berechnen Sie die Kosten des Algorithmus im EKM und im LKM.