Übungen zur Vorlesung
Einführung in die Algorithmik WS 15/16

Moritz Eissenhauer

Hasso-Plattner-Institut
Potsdam
24. November 2015

Lösung zu Übungsblatt 6

Lösung zu Aufgabe 2

a)

ZZ. Sort ist Korrekt (für $j > i \geq 0$)

Beweis. Seien x und y beliebige Elemente in A mit $i \leq ind(x) \leq j$ und $i \leq ind(y) \leq j$, wobei $ind(x)$ der Index von x in A ist.
Zeige: Falls $x < y$ dann ist nach Ausführung von sort $ind(x) < ind(y)$ ($\Leftrightarrow A$ ist sortiert von i bis j). (I)
Induktiv über $j - i$:
Induktions Vorraussetzung: Falls $j - i = 1$ dann ist $j = i + 1$ und der erste **if**-Block wird betreten. Wegen den Bedingungen oben ist entweder $A[i] = x \wedge A[j] = y$ oder $A[j] = x \wedge A[i] = y$. Für den ersten Fall ist $A[j] < A[i] = False$, nichts wird getauscht und $ind(x) < ind(y)$ gilt. Für den zweiten Fall ist $A[j] < A[i] = True$, $A[i]$ und $A[j]$ werden getauscht und $ind(x) < ind(y)$ gilt.
Dannach wird unmittelbar returnt, (I) gilt also für $j - i = 1$.
Induktions Behauptung: (I) gilt für $j - i \leq n$, $n \geq 1$ beliebig.
Induktions Schritt: Zeige (I) gilt für $j - i \leq n \Rightarrow$ (I) gilt für $j - i \leq n + 1$.
Sei $j - i = n + 1$.
Da $n \geq 1$ gilt $j = i + 1$ nicht und der **if**-Block wird nicht betreten. l wird auf $j - i + 1 = n + 1$ gesetzt. Das heißt $\lfloor \frac{l}{3} \rfloor > 1$ also ist $\lceil j - \frac{l}{3} \rceil - i \leq n$. Der Aufruf in Zeile 8 sortiert die Elemente i bis inklusive $\lceil j - \frac{l}{3} \rceil$.
Es gilt auch $j - \lfloor i + \frac{l}{3} \rfloor \leq n$, also sortiert Zeile 9 $\lfloor i + \frac{l}{3} \rfloor$ bis i. Das Teilarray von i bis exklusive $\lfloor i + \frac{l}{3} \rfloor$ ist also duch 8 sortiert und $\lfloor i + \frac{l}{3} \rfloor$ bis j durch 9. Insbesondere gilt auch das alle Elemente im Teilarray $j - \lfloor i + \frac{l}{3} \rfloor$ bis j größer sind als alle anderen.
Mit Zeile 10 wird wieder i bis $j - \lfloor i + \frac{l}{3} \rfloor$ sortiert, das heißt das gesamte Array ist sortiert. $\square$

b)

Für $j - i = n$ lässt sich die Laufzeit als

$$\begin{aligned} T(1) &= c \\ T(n) &= 3T(\frac{2n}{3}) + c \end{aligned} \tag{1}$$

Mit dem Mastertheorem Fall a erhällt man:

$$\begin{aligned} &\log_b a > 2 \\ &f(n) = c \in \mathcal{O}(n^2) \\ \Rightarrow &T(n) \in \Theta(n^{\log_b a}) \\ \Leftrightarrow &T(n) \in \Theta(n^{\log_{1,5} 3}) \end{aligned} \tag{2}$$

Lösung zu Aufgabe 3

a)

ZZ. Genericsort ist korrekt

Beweis. Für x, y beiebige ganze Dezimalzahlen mit k (beliebig, fest) stellen gilt

$$x \neq y \Rightarrow \exists 1 \leq j \leq k : x_j \neq y_j \tag{3}$$

Für das kleinste dieser j gilt dann $x_j < y_j \Rightarrow: x < y$. Da die Zahlen in absteigender Reihenfolge sortiert werden und das Sortierverfahren X stabil ist, steht $x \in A$ genau dann vor $x \neq y \in A$ wenn für die kleinste Stelle j an der sich x und y unterscheiden $x_j < y_j$ gilt. □

b)

Nach der ersten for-Schleife stehen in B am index i die Anzahl der Vorkommen von i in A. Nach der zweiten schleife steht am Index i Anzahl der Vorkommen aller $x \leq i$ in A minus eins. Das is Äquivalent zu dem jeweils letzten Index im sortierten array, der den wert i hat.
Der Algorithmus sortiert auch korrekt (aber nicht stabil) wenn die Schleife aufsteigend läuft weil nur die reihenfolge geändert wird wann welcher Wert in C geschrieben wird, aber trotzdem Die gleichen Werte an die entsprechenden Stellen geschrieben werden.

c)

Anmerkung: Es is schwierig die Reienfolge der elemente von der der Schlüssel zu unterscheiden, wenn die Elemente selbst die Schlüssel sind.

ZZ. Linearsort ist stabil

Beweis. In der dritten Schleife wird ein wert x in C zuerst and die letzte stelle aller x im sortierten array geschrieben.
Wenn das erste mal der Wert x in A gelesen wird wird er also an die Letzte Stelle aller x in C geschrieben. Der Index an den das nächste x geschrieben wird wird dann um eins verringert. Da die Schleife rückwärts läuft wird also immer das n-letzte x an die n-letzte Stelle der aller x geschrieben. Die reihenfolge bleibt also gleich. □

d)

ZZ. Es gibt im worst case $\Omega(n^2)$ Vertauschungen.

Beweis. Wenn für alle $i < \frac{n}{2}$ gilt $A[2i] > A[2i+1]$ dann wird zuerst beim ersten Durchlauf der äuseren Schleife bei jedem zweiten Element getauscht, also werden $\frac{n}{2}$ Vertauschungen vorgenommen. Beim jedem weiteren Durchgang wird jeweils eine Vertauschung weniger vorgenommen. Die Beschriiebene Form lässt sich für alle $l > 1$ und n herstellen.
Die gesamtzahl der Vertauschungen ist also $n\frac{n}{2} \in \Omega(n^2)$. □

e)
Die Laufzeit von Genericsort mit Bubblesort als Subroutine X ist $T_G(k,n) = kT_{Bubble}(n)$ wobei T_{Bubble} die Laufzeit von Bubblesort ist. $T_G(k,n)$ ist somit ein $\Theta(kn^2)$.

f)

Die Lufzeit von Linearsort ist $f_1(n) + f_2(z) + f_3(n) + c$ wobei f_i die Laufzeit für die i-te Schleife ist. f_1 und f_3 sind in $\Theta(n)$, f_2 ist in $\Theta(z)$. Damit ist die gesamtlaufzeit $\Theta(n+z)$.
ZZ. Für $z = cn$ gilt die Laufzeit ist in $O(n)$

Beweis. Die Laufzeit für z beliebig ist in $\Theta(n+z)$, die Laufzeit für $z = cn$ ist damit in $\Theta(n + cn) = \Theta(n) \cup \mathcal{O}(n)$ $\square$

g)
Die Laufzeit für Genericsort mit Linearsort als Subroutine ist $T_G(k,n) = kT_{Linear}(n,z)$ mit $z = 10$. $T_G(k,n)$ ist also in $\Theta(kn)$.