Matthias Krause
Alexander Moch

Mannheim, den 19.10.2020

CS 307 Algorithmen und Datenstrukturen, Herbstsemester 2020

Lösungsskizze zum Übungsblatt 4

AUFGABE 4.1:
Wenden Sie Counting Sort auf die Eingabe $A = (1, 6, 0, 3, 2, 0, 6, 5, 4, 2, 3, 3, 0, 4, 5, 6, 1, 1)$ an ($n = 18$, $k = 6$) und beantworten Sie folgende Fragen:

a) Was ist der Inhalt von C nach Schritt 3?

$h(0) = 3$, $h(1) = 3$, $h(2) = 2$, $h(3) = 3$, $h(4) = 2$, $h(5) = 2$, $h(6) = 3$

$\Rightarrow H(0) = 3$, $H(1) = 6$, $H(2) = 8$, $H(3) = 11$, $H(4) = 13$, $H(5) = 15$, $H(6) = 18$
(Inhalt von C)

b) Welche Permutation $\pi \in \mathfrak{S}_{18}$ beschreibt die Positionsänderungen der Elemente der Eingabe A hin zur sortierten Liste B nach Beendigung des Algorithmus? (Formal: Geben Sie jene Permutation $\pi \in \mathfrak{S}_{18}$ an, die $A[i] = B[\pi(i)]$ für $i = 1, \ldots, 18$ erfüllt.)

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 4 & 16 & 1 & 9 & 7 & 2 & 17 & 14 & 12 & 8 & 10 & 11 & 3 & 13 & 15 & 18 & 5 & 6 \end{pmatrix}$$

AUFGABE 4.2:
Wir betrachten nun eine Modifikation von Counting Sort, bei der in Zeile 4

For $i \leftarrow n$ downto 1

durch

For $i \leftarrow 1$ to n

ersetzt sei. Beantworten Sie die folgenden Fragen (unter Angabe einer Begründung!):

a) Löst der modifizierte Algorithmus noch das Sortierproblem?

Ja, die Änderung in Zeile 4 des Algorithmus wirkt sich lediglich auf die Reihenfolge der Elemente innerhalb jedes Teilfeldes gleicher Werte in B aus. (Details: vgl. (b))

b) Ist das entstehende Verfahren stabil?

Nein. Fixiere beliebiges $i \in \{1, \ldots, k\}$:

- *Seien $1 \leq r_1 \leq \ldots \leq r_{h(i)} \leq n$ die Positionen r mit $A[r] = i$.*
- *Sei $H(i-1)+1, \ldots, H(i)$ die Position, in die die Elemente $A[r_1] \ldots A[r_{h(i)}]$ kopiert werden müssen in Feld B.*

Dann wird

- *$A[r_{h(i)}]$ auf $H(i-1)+1$,*

– $A\left[r_{h(i)-1}\right]$ *auf* $H\,(i-1)+2$,

– $\vdots$

– $A\,[r_1]$ *auf* $H(i)$

kopiert, also seitenverkehrt; d.h. nicht stabil.

AUFGABE 4.3:

Ein Sortierverfahren heißt stabil, wenn in dessen Ausgabe die Objekte mit gleichem Schlüssel in der selben Reihenfolge wie in der Eingabe vorkommen.

a) Betrachten Sie den folgenden Sortieralgorithmus in Pseudocode:

```
algorithm SELECTION-SORT (A, n)
1    for i := 1 to n − 1
2        i_min := i;
3        for j := i + 1 to n
4            if (A[j] < A[i_min])
5                i_min := j;
6        swap(A[i], A[i_min]); //vertauscht die beiden Einträge
```

Ist SELECTION-SORT stabil?

Nein; betrachte zum Beispiel den Fall $A = [2, 2, 1]$.

b) Geben Sie eine einfache Methode an, mit der man aus einem beliebigen, allgemeinen Sortieralgorithmus einen stabilen Sortieralgorithmus erzeugen kann.
Hinweis: Die Schlüsselmenge und die Ordnung darauf dürfen modifiziert werden. *Angenommen, die Schlüssel* $k_1, \ldots, k_n$ *stammen aus einer Menge* K *mit der strikten Ordnung* $\prec$. *Für jedes Objekt* i *führe einen neuen Schlüssel* (k_i, i) *ein und definiere die neue Ordnung* $\prec'$ *gemäß* $(k_i, i) \prec' (k_j, j) \Leftrightarrow k_i \prec k_j \vee (k_i = k_j \wedge i < j)$.

AUFGABE 4.4$^{\mathbf{K}}$:

In dieser Aufgabe betrachten wir Klassen $\mathcal{H}$ von Hashfunktionen $h : U \to \{0, \ldots, m-1\}$ und eine zufällig (gemäß Gleichverteilung) gewählte Hashfunktion $h \in \mathcal{H}$.

a) Wann sagen wir, $\mathcal{H}$ ist eine universelle Klasse?

Wenn für beliebige Schlüssel $x \neq y$ *aus* U *gilt:* $Pr\{h(x) = h(y)\} = 1/m$.

b) Wir sagen, $\mathcal{H}$ ist 2-unabhängig, wenn für alle $x, y \in U$ mit $x \neq y$ und alle $i, j \in \{0, \ldots, m-1\}$ gilt: $Pr\{h(x) = i \ \wedge \ h(y) = j\} = 1/m^2$.
Beweisen Sie oder finden Sie ein Gegenbeispiel: Wenn $\mathcal{H}$ 2-unabhängig ist, dann ist $\mathcal{H}$ auch universell.

Beweis: Sei $x \neq y$. $Pr\{h(x) = h(y)\} = \sum_{i=0}^{m-1} Pr\{h(x) = i \ \wedge \ h(y) = i\} = \sum_{i=0}^{m-1} 1/m^2 = 1/m$.

c) Beweisen Sie oder finden Sie ein Gegenbeispiel: Wenn $\mathcal{H}$ universell ist, dann ist $\mathcal{H}$ auch 2-unabhängig.

Gegenbeispiel: Betrachte die aus der Vorlesung bekannte universelle Klasse von Hashfunktionen $h_a(x) = \sum_{l=1}^{r} a_l x_l \mod m$ *(für eine Primzahl* m*). Sei* $y = 2x$ *und* $j = 2i$ ($\mod m$). *Dann ist* $Pr\{h(x) = i \ \wedge \ h(y) = j\} = Pr\{\sum_{l=1}^{r} a_l x_l = i \pmod m\} = m^{r-1}/m^r \neq 1/m^2$.

AUFGABE 4.5:
Wir betrachten die Hashtabelle $T = T[0], \ldots, T[m-1]$, $m = 1000$, mit

$$h(k) = \lfloor m\,(k \cdot A \mod 1) \rfloor \text{ für } A = \frac{\sqrt{5}-1}{2}$$

(Multiplikationsmethode).
Wo werden die Schlüssel 61, 62, 63, 64, 65 in T abgelegt?

$$h(61) = 700,\ h(62) = 318,\ h(63) = 936,\ h(64) = 554,\ h(65) = 172$$