9 Sortieren in Linearzeit

9.1 Counting Sort

Algorithm 1: CountingSort

Input: Array der Länge n mit Einträgen $A[i] \in \{1, 2, \ldots, k\}$
Output: Sortiertes Array B

Let C be Array of length k
for $i \leftarrow 1$ **to** k **do**
    $C[i] \leftarrow 0$
end
for $j \leftarrow 1$ **to** n **do**
    $C[A[j]] \leftarrow C[A[j]] + 1$
end
for $i \leftarrow 1$ **to** $k - 1$ **do**
    $C[i+1] \leftarrow C[i+1] + C[i]$
end
Let B be Array of length n **for** $j \leftarrow n$ **to** 1 **do**
    $B[C[A[j]]] \leftarrow A[j]$
    $C[A[j]] \leftarrow C[A[j]] - 1$
end

Satz 9.1. *CountingSort arbeitet korrekt und besitzt Laufzeitfunktion in* $\Theta(n + k)$.

9.2 RadixSort

Algorithm 2: RadixSort

Input: Array der Länge n. Jeder Eintrag ist d-stellige Zahl, wobei die erste Stelle die niederste und die letzte Stelle die höchste ist.
Output: Sortiertes Array.

1 **for** $i \leftarrow 1$ **to** d **do**
2   sortiere A nach i-ter Stelle mit stabilem Sortierverfahren.
3 **end**

Satz 9.2. *Für* n *Zahlen mit je* d *Stellen, bei denen jede Stelle bis zu* k *mögliche Werte annehmen kann, sortiert RadixSort diese Zahlen korrekt in Zeit* $\Theta\big(d(n + k)\big)$*, falls das stabile Sortierverfahren in Zeile* 2 *Laufzeit* $\Theta(n + k)$ *besitzt (bspw. CountingSort).*

9.3 BucketSort

Algorithm 3: BucketSort

Input: Array A der Länge n mit Einträgen $A[i] \in (0, 1]$
Output: Sortierte Liste

Let B be Array of length n **for** $i \leftarrow 1$ **to** n **do**
  $B[i] \leftarrow [\ ]$
end
for $i \leftarrow 1$ **to** n **do**
  insert $A[i]$ into list $B\big[\lceil n \cdot A[i] \rceil\big]$
end
for $i = 1$ **to** n **do**
  sort list $B[i]$ with insertionSort
end
concatenate lists $B[1], B[2], \dots, B[n]$

Satz 9.3. *Werden die Einträge unabhängig und gleichverteilt aus* $(0, 1]$ *gezogen, so ist die erwartete Laufzeit von BucketSort in* $\mathcal{O}(n)$.

Bemerkung.
- Die erwartete Laufzeit von BucketSort ist linear in n, da die erwartete Summe der Quadrate der Bucketgrößen linear in n ist.
- BucketSort kann durch Skalierung für beliebige Zahlenbereiche angewandt werden.

Laufzeitbeweis. Es sei n die Anzahl der Elemente in Bucket $B[i], i = 1, \dots, n$. Dann gilt für die Laufzeitfunktion $T(n)$ von BucketSort:

$$T(n) \leq c \cdot n + \sum_{i=1}^{n} d \cdot n_i^2$$

für Konstanten $c, d > 0$. Wegen Linearität des Erwartungswertes gilt für die erwartete Laufzeit:

$$E[T(n)] \leq c \cdot n + \sum_{i=1}^{n} d \cdot E[n_i^2]$$

Definiere Zufallsvariablen

$$x_{ij} \leq \begin{cases} 1, & \text{falls } A[j] \text{ in Bucket } B[i] \text{ landet.} \\ 0, & \text{sonst.} \end{cases}$$

Dann ist

$$n_i = \sum_{j=1}^{n} x_{ij}$$

und

$$\begin{aligned}
E[n_i^2] &= E\Big[\Big(\sum_{j=1}^{n} x_{ij}\Big)\Big] \\
&= E\Big[\sum_{j=1}^{n}\sum_{k=1}^{n} x_{ij}x_{ik}\Big] \\
&= E\Big[\sum_{j=1}^{n} x_{ij}^2 + \sum_{j=1}^{n}\sum_{k\neq j} x_{ij}x_{ik}\Big] \\
&= \sum_{j=1}^{n} E[x_{ij}^2] + \sum_{j=1}^{n}\sum_{k\neq j} E[x_{ij}x_{ik}] \\
&= \sum_{j=1}^{k} \frac{1}{n} + \sum_{j=1}^{n}\sum_{k\neq j} E[x_{ij}]E[x_{ik}] \\
&= 1 + \frac{n(n-1)}{n^2} = 2 - \frac{1}{n}
\end{aligned}$$

□