Teilaufgabe IV

Es sei $A[0 \dots n-1]$ ein Array von paarweise verschiedenen ganzen Zahlen.

Wir interessieren uns für die Zahl der Inversionen von A; das sind Paare von Indices (i, j), sodass $i < j$ aber $A[i] > A[j]$. Die Inversionen im Array $[2, 3, 8, 6, 1]$ sind $(0, 4)$, da $A[0] > A[4]$ und weiter $(1, 4)$, $(2, 3)$, $(2, 4)$, $(3, 4)$. Es gibt also 5 Inversionen.

(a) Wie viel Inversionen hat das Array $[3, 7, 1, 4, 5, 9, 2]$?

> - $(0, 1)$: $3 > 1$
> - $(0, 6)$: $3 > 2$
> - $(1, 2)$: $7 > 1$
> - $(1, 3)$: $7 > 4$
> - $(1, 4)$: $7 > 5$
> - $(1, 6)$: $7 > 2$
> - $(3, 6)$: $4 > 2$
> - $(4, 6)$: $5 > 2$

(b) Welches Array mit den Einträgen $\{1, \dots, n\}$ hat die meisten Inversionen, welches hat die wenigsten?

> Folgt nach der 1 eine absteigend sortiere Folge, so hat sie am meisten Inversionen, z. B. $\{1, 7, 6, 5, 4, 3, 2\}$. Eine aufsteigend sortierte Zahlenfolge hat keine Inversionen, z. B. $\{1, 2, 3, 4, 5, 6, 7\}$.

(c) Entwerfen Sie eine Prozedur `int merge(int[] a, int i, int h, int j);`

welche das Teilarray a[i.,j] sortiert und die Zahl der in ihm enthaltenen Inversionen zurückliefert, wobei die folgenden Vorbedingungen angenommen werden:

- $0 \leq i \leq h \leq j < n$, wobei n die Länge von a ist (`n = a.length`).
- $a[i \dots h]$ und $a[h + 1 \dots j]$ sind aufsteigend sortiert.
- Die Einträge von $a[i \dots j]$ sind paarweise verschieden.

Ihre Prozedur soll in linearer Zeit, also $\mathcal{O}(j - i)$ laufen. Orientieren Sie sich bei Ihrer Lösung an der Mischoperation des bekannten Mergesort-Verfahrens.

(d) Entwerfen Sie nun ein Divide-and-Conquer-Verfahren zur Bestimmung der Zahl der Inversionen, indem Sie angelehnt an das Mergesort-Verfahren einen Algorithmus ZI beschreiben, der ein gegebenes Array in sortierter Form liefert und gleichzeitig dessen Inversionsanzahl berechnet. Im Beispiel wäre also

$$ZI([2, 3, 8, 6, 1]) = ([1, 2, 3, 6, 8], 5)$$

Die Laufzeit Ihres Algorithmus auf einem Array der Größe n soll $\mathcal{O}(n \log(n))$ sein.

Sie dürfen die Hilfsprozedur merge aus dem vorherigen Aufgabenteil verwenden, auch, wenn Sie diese nicht gelöst haben.

(e) Begründen Sie, dass Ihr Algorithmus die Laufzeit $\mathcal{O}(n \log(n))$ hat.

(f) Geben Sie die Lösungen folgender asymptotischer Rekurrenzen (in O-Notation) an:

(i) $T(n) = 2 \cdot T(\frac{n}{2}) + \mathcal{O}(\log n)$

(ii) $T(n) = 2 \cdot T(\frac{n}{2}) + \mathcal{O}(n^2)$

(iii) $T(n) = 3 \cdot T(\frac{n}{2}) + \mathcal{O}(n)$