ETH Zürich
Institut für Theoretische Informatik
Prof. Dr. Angelika Steger, Dr. Johannes Lengler
Übungsleitung: Florian Meier

HS 2019
Abgabe 15.10.2019

Algorithmen und Komplexität
Übungsblatt 4

Peer-Grading: Aufgabe 1

Korrektur durch die Assistenten: Aufgaben 2 und 3

⋆ ⋆ ⋆

Aufgabe 1

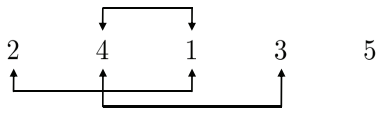
a) Gegeben sei ein aufsteigend sortiertes Array $a[1..n]$ mit n Zahlen, wobei einzelne Zahlen mehrfach enthalten sein können. Entwerfen Sie einen Algorithmus, der für ein gegebenes x in Laufzeit $\mathcal{O}(\log n)$ herausfindet, wie oft x im Array a vorkommt. Zeigen Sie Korrektheit und dass die Laufzeit eingehalten wird.

b) Gegeben seien zwei *sortierte* Arrays $a[1..n]$ und $b[1..n]$ mit jeweils n Zahlen, wobei keine Zahl doppelt vorkommt. Entwerfen Sie einen Algorithmus, der in Laufzeit $\mathcal{O}(\log n)$ den Median aller $2n$ Zahlen findet. Zeigen Sie Korrektheit und dass die Laufzeit eingehalten wird.

Aufgabe 2

Sei $(a_1, a_2, \ldots, a_n)$ eine Liste von paarweise unterschiedlichen Zahlen aus $\{1, 2, \ldots, n\}$. Das Paar (i, j) ist eine *Inversion* falls $i < j$ und $a_i > a_j$. Das Ziel ist die Anzahl der Inversionen in der Liste zu berechnen.

Diese Anzahl misst, wie weit die Liste $(a_1, \ldots, a_n)$ von der sortierten Liste $(1, 2, \ldots, n)$ entfernt ist. Zum Beispiel gibt es in der folgenden Liste 3 Inversionen: $(1, 3)$, $(2, 3)$ und $(2, 4)$.



(a) Wie gross ist die Anzahl Inversionen in einer Liste $(a_1, \ldots, a_n)$ minimal und maximal? Geben Sie für beide Fälle je ein Beispiel.

(b) Beschreiben Sie (kurz und in Worten) einen einfachen Algorithmus, der die Inversionen zählt und in Zeit $O(n^2)$ läuft. Begründen Sie die Laufzeit.

Wir entwerfen einen schnelleren Algorithmus, der die Anzahl Inversionen in Zeit $O(n \log n)$ berechnet. Dazu soll der Mergesort Algorithmus verwendet werden.

(c) Zuerst betrachten wir eine Liste L in der die Elemente in der linken und der rechten Hälfte jeweils aufsteigend sortiert sind. Wie kann man den Merge-Schritt von Mergesort so erweitern, dass er aus den beiden Hälften eine sortierte Liste erstellt und gleichzeitig die Anzahl der Inversionen von L zählt? Achten Sie darauf, dass der Merge-Schritt weiterhin in Zeit $O(|L|)$ läuft, wobei $|L|$ die Länge der betrachtenten Liste L bezeichnet.

(d) Verwenden Sie (c), um den Mergesort Algorithmus so zu erweitern, dass er die Anzahl der Inversionen einer beliebigen Liste zählt. Der Algorithmus soll in Zeit $O(n \log n)$ laufen. Begründen Sie die Korrektheit und Laufzeit des Algorithmus.

Aufgabe 3

Gegeben ein Array A bestehend aus n Zahlen $a_1, a_2, \ldots, a_n$. Wir nennen eine Zahl x omnipräsent in A, falls x mindestens $\frac{n+1}{2}$ mal in A vorkommt. Der Vergleichsoperator SAME bekommt zwei Zahlen a_i und a_j als Input und gibt eine 1 aus falls $a_i = a_j$ und er gibt eine 0 aus falls $a_i \neq a_j$. Entwerfen Sie einen Algorithmus, der mit $\mathcal{O}(n)$ Vergleichen (d.h. mit $\mathcal{O}(n)$ Aufrufen des SAME Operators) entscheidet, ob es eine Zahl omnipräsente Zahl x in A gibt und dieses x ausgibt, falls es existiert.

Gehen Sie dabei folgendermassen vor.

a) Nehmen Sie an, dass n gerade ist. Wie kann mit $\mathcal{O}(n)$ Vergleichen aus dem Array A ein Array B mit Länge $n' \leq \frac{n}{2}$ erzeugt werden, sodass falls eine Zahl x omnipräsent in A ist, sie auch omnipräsent in B ist?

b) Nehmen Sie an, dass $n = 2m + 1$ ungerade ist. Wie können Sie mit $\mathcal{O}(n)$ Vergleichen entweder die omnipräsente Zahl x finden, oder das Problem wie in a) auf ein Array der Länge höchstens m reduzieren?

c) Wenden Sie die Algorithmen aus a) und b) rekursiv an und analysieren Sie die Laufzeit.

ABGABE DER HAUSAUFGABEN IN DER VORLESUNG AM 15.10.2019.