Lehrstuhl für Informatik I
Algorithmen, Komplexität und wissensbasierte Systeme
Universität Würzburg

Würzburg, den 8. November 2018

Prof. Dr. Alexander Wolff
Johannes Zink, M. Sc.

4. Übungsblatt zur Vorlesung Algorithmen und Datenstrukturen (Winter 2018/19)

Aufgabe 1 – Heaps

a) Wo kann sich das kleinste Element in einem MaxHeap befinden, wenn alle Elemente verschieden sind? **1 Punkt**

b) Ist die Folge $\langle 18, 17, 14, 14, 11, 10, 12, 14, 11, 12, 11\rangle$ ein MaxHeap? **1 Punkt**

c) Was bewirkt $\texttt{MaxHeapify}(A, i)$, wenn $A[i]$ größer als seine Kinder ist? **1 Punkt**

d) Zeigen Sie, dass die maximale Laufzeit von MaxHeapify über alle Felder der Länge n in $\Omega(\log n)$ ist. Dazu genügt es (für beliebig große, aber nicht notwendigerweise für alle $n \in \mathbb{N}$) ein konkretes Feld A_n der Größe n und ein Element $1 \leq i \leq n$ anzugeben, so dass $\texttt{MaxHeapify}(A_n, i)$ tatsächlich $\Omega(\log n)$ Zeit benötigt. **1 Punkt**

Aufgabe 2 – Nachbartupel

Im Folgenden sei das Feld A eine zufällige Permutation von $\langle 1, \ldots, n\rangle$, wobei n gerade sei.

Ein Tupel (i, j) mit $1 \leq i < j \leq n$ heißt *passend*, wenn $A[i] + A[j] = n + 1$ gilt. Ein Tupel (i, j) mit $1 \leq i < j \leq n$ und $j = i + 1$ heißt *benachbart*.

Verwenden Sie im Folgenden Zufallsvariable und Indikator-Zufallsvariable, geben Sie die einzelnen Rechenschritte an und vereinfachen Sie das Endergebnis so weit wie möglich.

a) Was ist der Erwartungswert für die Anzahl passender Tupel, die benachbart sind? **3 Punkte**

b) Was ist der Erwartungswert für die Anzahl passender Tupel? **3 Punkte**

c) Was ist der Erwartungswert für die Anzahl passender Tupel (i, j) mit der Eigenschaft, dass $A[i] \leq i$? **3 Punkte**

Aufgabe 3 – MinHeaps

Gegeben sei eine Menge $S \subseteq \mathbb{Z}$ von n Zahlen und eine Zahl $z \in \mathbb{Z}$. Ein Algorithmus soll alle Zahlen in S ausgeben, die echt kleiner als z sind.

a) Zeigen Sie: jeder Algorithmus für dieses Problem braucht im Worst-Case $\Omega(n)$ Zeit. **2 Punkte**

b) Geben Sie in Pseudocode einen Algorithmus an, der dieses Problem auf einem MinHeap S löst. Der Algorithmus soll echt schneller als linear in n sein, wenn die Mächtigkeit k der Ergebnismenge klein ist (das heißt $k \in o(n)$). Am Ende soll S immer noch ein MinHeap sein, der die gleichen Elemente wie am Anfang enthält. Hierfür können 3 Punkte erreicht werden. Es gibt den 4. Punkt, wenn die Laufzeit nicht von n, sondern nur von k abhängt. **4 Punkte**

c) Geben Sie die Laufzeit Ihres Algorithmus aus Aufgabenteil b) in Abhängigkeit von n und k an. **1 Punkt**

Bitte werfen Sie Ihre Lösungen bis **Donnerstag, 15. November 2018, 13:00 Uhr** in den Vorlesungs-Briefkasten im Informatik-Gebäude. Geben Sie stets die Namen und Übungsgruppen aller BearbeiterInnen sowie die Übungsgruppe, in der das Blatt zurückgegeben werden soll, an.

Grundsätzlich sind stets alle Ihrer Aussagen zu begründen und Ihr Pseudocode ist stets zu kommentieren.

Die Lösungen zu den mit PABS gekennzeichneten Aufgaben, geben Sie bitte nur über das PABS-System ab. Vermerken Sie auf Ihrem Übungsblatt, in welchem Repository (*sXXXXXX*-Nummer) die Abgabe zu finden ist. Geben Sie Ihre Namen hier als Kommentare in den Quelltextdateien an.