Algorithmen & Datenstrukturen 2

Wintersemester 2022/23

Prof. Dr. Martin Hoefer
Tim Koglin, Marco Schmalhofer

GOETHE UNIVERSITÄT FRANKFURT AM MAIN
Institut für Informatik
Algorithmen und Komplexität

Übungsblatt 2

Ausgabe: 01.11.2022
Abgabe: 08.11.2022, **08:00**

Aufgabe 2.1 *Auswahlproblem* (4 Punkte)

Betrachten Sie den in der Vorlesung vorgestellten Algorithmus zur Lösung des Auswahlproblems.

Konstruieren Sie eine Folge der Länge 5, so dass der Algorithmus bei Verwendung der Pivotfunktion $pivot(links, rechts) = links$ auf der Suche nach dem *viert*kleinsten Schlüssel die Problemgröße stets nur um 1 verringert. Der Algorithmus soll insgesamt also fünf Schritte benötigen, bis er terminiert. Wenden Sie zum Beweis der Korrektheit den Algorithmus auf Ihre Folge an.

Aufgabe 2.2 *Mergesort auf Externspeichern* (3 + 3 Punkte)

Wir betrachten das verbesserte Externspeichersortieren: Sei M die Größe des Hauptspeichers und $B = M^{1/4}$ die Blockgröße, die bei einem Lesezugriff auf den Externspeicher gelesen werden kann. Sei $n = M^{16}$ die Größe der zu sortierenden Folge.

a) Wie viele Mergephasen benötigt das Sortieren? Geben Sie die Größenordnung in Θ-Notation als Konstante oder in Abhängigkeit von M sowie den Rechenweg an.

b) Wie viele Festplattenzugriffe sind notwendig? Geben Sie die Größenordnung in Θ-Notation als Konstante oder in Abhängigkeit von M sowie den Rechenweg an.

Hinweis: Beachten Sie, dass jeweils auch eine untere Schranke argumentiert werden muss.

Aufgabe 2.3 *Hybride Sortierverfahren* (5 + 5 Punkte)

Betrachten Sie folgende Abwandlung von Mergesort: Anstatt bis zur Intervallgröße `rechts-links<=1` rekursiv vorzugehen, soll Insertion Sort verwendet werden, sobald die Größe des Teilproblems einen Schwellenwert k nicht mehr überschreitet.

Bestimmen Sie jeweils die asymptotische *best*- sowie *worst-case* Laufzeit des resultierenden Verfahrens in Abhängigkeit von n in Θ-Notation für folgende Schwellenwerte:

a) $k = \log_2 n$.

b) $k = \sqrt{n}$.

Hinweis: Sie können vereinfachend annehmen, dass $n = 2^{(2^r)}$ für ein $r \in \mathbb{N}$.

Aufgabe 2.4 *Teilweise sortiert* (5 + 5 Punkte)

Sei A ein Array mit n Elementen A[1],...,A[n], welches paarweise verschiedene Zahlen speichert. Weiterhin ist jeweils die folgende Eigenschaft bekannt:

a) Jedes Element ist maximal 2022 Stellen von seiner Position in der sortierten Folge entfernt.

b) Jedes Element ist maximal $\sqrt{n}$ Stellen von seiner Position in der sortierten Folge entfernt. Sie können annehmen, dass $n = r^2$ für ein $r \in \mathbb{N}$.

Entwerfen Sie für die Fälle a) und b) jeweils ein Sortierverfahren, welches A in worst-case Laufzeit $\mathcal{O}(n)$ sortiert, oder zeigen Sie durch Anpassen des Arguments aus der Vorlesung (Folien 65-69), dass es kein solches gibt.

Aufgabe 2.5 *Guest-Scheduling* (10 Punkte)

Ein Hotel möchte n alleinreisende Gäste in gegebenen festen Zeiträumen beherbergen. Die Beherbergung eines Gastes $i \in \{1, \ldots, n\}$ beginnt mit dem Check-in am Tag $s_i \geq 0$ und endet mit dem Check-out am Tag $t_i > s_i$. Dabei sind alle $2n$ Tage s_i, t_i, $i = 1, \ldots, n$, unterschiedlich. Das Hotel verfügt über 50 Einzelzimmer.

Entwickeln Sie einen Algorithmus, der in Zeit $\mathcal{O}(n \log n)$ korrekt entscheidet, ob das Hotel alle n Gäste beherbergen kann. Beschreiben Sie zuerst Ihre Idee und geben Sie danach Ihren Algorithmus in Pseudocode an. Aus der Vorlesung bekannte Verfahren und deren Eigenschaften müssen Sie dabei nicht nochmal in Pseudocode aufschreiben, sondern lediglich benennen. Begründen Sie anschließend die Korrektheit und warum Ihr Algorithmus die vorgegebene Laufzeitschranke einhält.

Bei allgemeinen Anmerkungen zu den Übungsaufgaben oder Fragen zum Übungsbetrieb erreichen Sie uns unter der folgenden E-Mail-Adresse: algo222@cs.uni-frankfurt.de.