11. Aufgabenblatt zur Vorlesung

Algorithmen, Datenstrukturen und Datenabstraktion WS 17/18

Wolfgang Mulzer, Katharina Klost

Abgabe am 19. Januar 2018 bis 10 Uhr in die jeweiligen Tutorenfächer

Aufgabe 1 $(2,3)$-Bäume *10 Punkte*

(a) Fügen Sie die Schlüssel A, L, P, D, R, E, I, X, Y, Z in dieser Reihenfolge in einen anfangs leeren $(2,3)$-Baum ein. Zeigen Sie die einzelnen Schritte. Löschen Sie sodann die Schlüssel A, L, X.

(b) Beschreiben Sie schematisch einen $(2,3)$-Baum mit n Schlüsseln, so dass eine Löschoperation $\Omega(\log n)$ Rebalancierungsoperationen benötigt.

Aufgabe 2 (a,b)-Bäume *10 Punkte*

(a) Beschreiben Sie, wie man in einem (a,b)-Baum mit n Schlüsseln die Operation `findRange`(k_1, k_2) implementieren kann, die alle Schlüssel k liefert, für die $k_1 \leq k \leq k_2$ ist. Die Laufzeit soll $O(b \log_a n + s)$ betragen. Dabei ist s die Anzahl der gelieferten Schlüssel.

(b) Seien T_1 und T_2 zwei (a,b)-Bäume, und sei S_1 die Schlüsselmenge von T_1 und S_2 die Schlüsselmenge von T_2. Sei x ein weiterer Schlüssel. Alle Schlüssel in S_1 sind kleiner als x, und alle Schlüssel in S_2 sind größer als x Beschreiben Sie eine Operation `join`, die aus T_1, T_2 und x einen (a,b)-Baum für die Schlüsselmenge $S_1 \cup \{x\} \cup S_2$ erzeugt. Die Laufzeit sollte $O(b \log_a \max\{|S_1|, |S_2|\})$ betragen.

Hinweis: Betrachten Sie zunächst den Fall, dass T_1 und T_2 die gleiche Höhe haben. Achten Sie darauf, dass hinterher die (a,b)-Baum-Eigenschaften wieder hergestellt werden.

Aufgabe 3 Gieriges Münzwechseln *10 Punkte*

Betrachten Sie das Problem des Münzwechselns an der Supermarktkasse. Es soll ein Betrag von n Cents herausgegeben werden, wobei die minimale Anzahl an Münzen verwendet werden soll. Zur Erinnerung: es gibt Euro-Münzen für 1, 2, 5, 10, 20, 50, 100 und 200 Cent.

(a) Beschreiben Sie einen gierigen Algorithmus für das Problem, und implementieren Sie ihn in Java. Was ist die Laufzeit des Algorithmus?

(b) Beweisen Sie, dass Ihr Algorithmus aus (a) eine optimale Lösung liefert.

Hinweis: Machen Sie sich Gedanken über die Struktur einer optimalen Lösung: Wie viele 1-Cent Münzen kann eine optimale Lösung höchstens enthalten? Wie

viele 2-Cent Münzen? usw? Was ist der maximale Teilbetrag, der in einer optimalen Lösung mit Münzen mit Denomination höchstens c dargestellt werden kann, für $c \in \{2, 5, 10, 20, 50, 100\}$?

(c) Geben Sie ein Beispiel für eine Familie von Münzdenominationen, so dass der gierige Algorithmus nicht optimal ist. Dabei sollte es eine 1-Cent Münze geben, so dass sich jeder Betrag auch wechseln lässt.

Hinweis: Bitte formatieren Sie Ihre Abgaben mit LaTeX oder einem vergleichbaren elektronischen Textverarbeitungssystem. Ansonsten droht der Abzug von bis zu 10% der erreichbaren Punkte.