Effiziente Algorithmen

Sommersemester 2019

Prof. Dr. Martin Hoefer
Dr. Daniel Schmand
Martin Ludwig, Conrad Schecker

GOETHE UNIVERSITÄT
FRANKFURT AM MAIN

Institut für Informatik
Algorithmen & Komplexität

Übung 10

Ausgabe: 25.06.2019
Abgabe: 02.07.2019, 10:15

Aufgabe 10.1. (4 Punkte)

Eine dynamische Hash-Tabelle ändert die Tabellengröße nach der folgenden Regel:

Anfänglich ist die Tabellengröße $m = 1$. Bei Auslastungsfaktor $\lambda \geqslant \frac{2}{3}$ wird die Tabellengröße verdoppelt ($m_{neu} := 2m$), und bei Auslastungsfaktor $\lambda \leqslant \frac{1}{6}$ wird diese halbiert ($m_{neu} := \lceil \frac{1}{2} m \rceil$). Wir nehmen an, dass die tatsächliche Laufzeit einer `Insert()`- oder `Remove()`-Operation 1 ist, wenn keine Reorganisation erforderlich ist, und m (alte Tabellengröße) sonst. Eine `Lookup()`-Operation dauert immer 1.

Bestimme die kleinstmögliche konstante amortisierte Laufzeit der drei oben genannten Operationen. Bitte beim Rechnen die Gauß-Klammern vernachlässigen.

Aufgabe 10.2. (2 + 3 + 2 Punkte)

a) Gibt es für jedes $n \in \mathbb{N}$ einen Baum mit n Knoten, der gleichzeitig die Heapeigenschaft erfüllt und ein binärer Suchbäum ist? Beweisen Sie die Antwort.

b) Muss bei der `DecreaseKey`-Operation bei Fibonacci-Heaps ein markierter Knoten weiterhin markiert bleiben,

   i) nachdem er abgehängt wurde?

   ii) nachdem einer seiner Vorfahren abgehängt wurde?

   Begründen Sie jeweils die Antwort.

c) Sei T_k ein Fibonacci-Baum, bei dem die Wurzel k Kinder hat. Geben Sie in Abhängigkeit von k die minimal mögliche Tiefe eines Blattes in T_k an. Beweisen Sie ihre Aussage.

Aufgabe 10.3. (4 Punkte)

Zeigen Sie, dass die Höhe eines Fibonacci-Baums in einem Fibonacci-Heap nicht durch $\mathcal{O}(\log n)$ beschränkt werden kann.

Tipp: Zeigen Sie, dass es zu jeder Zahl n eine Folge von Operationen gibt, die einen Fibonacci-Heap erzeugt, der nur aus einem linear entarteten Baum mit n Schlüsseln besteht. Benutzen Sie Induktion.
