Übungen zu Algorithmen
WSI für Informatik
Kaufmann/Bekos/Schneck

Sommersemester 2018
Universität Tübingen
01.06.2018

Übungsblatt 6

Abgabe bis 07.06.2018
Besprechung: 11.06.2018 – 14.06.2018

Aufgabe 1: AVL-Bäume (2 + 2 + 2 Punkte)

(a) Fügen Sie die Zahlen 63, 30, 36, 31, 12, 50, 35, 5, 27, 59, 43, 17 (in dieser Reihenfolge) in einen (zu Beginn leeren) AVL-Baum ein. Sie dürfen die Schritte, in denen nicht rotiert, sondern nur eingefügt wird, zusammenfassen.

(b) Löschen Sie die Zahlen 36, 43 wieder aus dem Baum. Geben Sie die Schritte an.

(c) Leiten Sie eine Eingabefolge von n Zahlen für einen AVL-Baum ab, die keine Rotationen erfordert.

Aufgabe 2: Binäre Suchbäume (2 + 2 + 2 Punkte)

(a) Gegeben ein Pfad P von der Wurzel zum Blatt eines binären Suchbaumes. Sei B die Menge der Knoten auf diesem Pfad, A alle Knoten links von P und C alle Knoten rechts von P. Gilt dann $a \leq b \leq c$ für alle $a \in A$, $b \in B$ und $c \in C$?

(b) Zeigen Sie: Hat ein Knoten in einem binären Suchbaum zwei Kinder, so hat sein Nachfolger kein linkes Kind und sein Vorgänger kein rechtes Kind. Gilt diese Aussage auch für ein Knoten mit nur einem Kind?

(c) Zeigen oder widerlegen Sie: Sei T ein binärer Suchbaum und seien x und y Elemente in T. Der binäre Suchbaum, der sich nach Löschen von x und y (in dieser Reihenfolge) ergibt ist derselbe Suchbaum, der sich nach Löschen von y und x (in dieser Reihenfolge) ergibt.

Aufgabe 3: Rot-Schwarz-Bäume (1 + 1 + 3 + 1 Punkte)

Rot-Schwarz-Bäume sind eine Möglichkeit, um balancierte Suchbäume zu implementieren. Wir betrachten nun binäre Suchbäume, deren Knoten folgende Felder enthalten: *color, key, left, right* und p (Parent). Alle Knoten werden als innere Knoten betrachtet, die Blätter durch Zeiger auf `nil` dargestellt. Ein binärer Suchbaum, der folgende Eigenschaften erfüllt, ist ein Rot-Schwarz-Baum:

E.1: Jeder Knoten ist schwarz oder rot.

E.2: Die Wurzel ist schwarz.

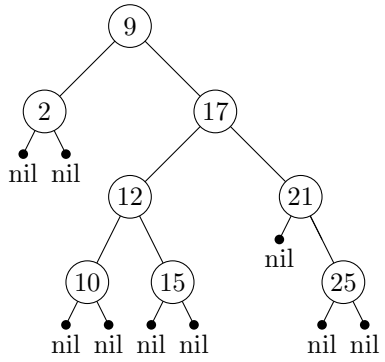
E.3: Jedes Blatt (`nil`) ist schwarz.

E.4: Falls ein Knoten rot ist, so sind beide Kinder schwarz.

E.5: Für jeden Knoten gilt: Alle Pfade zu den Nachfahren, die Blätter (`nil`) sind, haben die gleiche Anzahl von schwarzen Knoten.

(a) Färben Sie den angegebenen Suchbaum so, dass es sich um einen Rot-Schwarz-Baum handelt.

(b) In einem Rot-Schwarz-Baum ist das Einfügen von Elementen eine komplizierte Operation, die eventuell Rebalancierungen und Umfärbungen nach sich zieht. Fügen Sie zwei geeignete Elemente so in den Suchbaum ein, dass Ihre Färbung aus Teil (a) erhalten bleibt und trotzdem ein gültiger Rot-Schwarz-Baum entstehen.



(c) Zeigen Sie, dass ein Red-Black-Tree höchstens Tiefe $2\log(n+1)$ hat. Dabei ist n die Anzahl der (inneren) Knoten.

(d) Ist jeder Rot-Schwarz-Baum auch ein AVL-Baum? Begründen Sie Ihre Antwort.

Aufgabe 4: Implementation von Binären Suchbäumen (2 + 10 Punkte)

Laden Sie die Java-Vorlage aus dem Moodle herunter und implementieren Sie:

(a) Die Klasse `BSTreeNode`, die das gegebene Interface `TreeNode` implementiert.

(b) Die fünf Methoden der Klasse `BSTree`, die keine Implementierung haben.

Verwenden Sie bei Ihrer Implementierung sinnvolle Variablennamen und kommentieren Sie Ihren Code! Laden Sie Ihre Lösung ins Moodle. Nicht kompilierende Abgaben werden **mit 0 Punkten** bewertet.