6. Übungsblatt zu Grundzüge von Algorithmen und Datenstrukturen, WS 15/16

Prof. Markus Bläser
http://www-cc.cs.uni-saarland.de/course/50/

Abgabe: 3. Dezember 2015, 12:00 Uhr

Aufgabe 6.1 a) Fügen Sie in einen leeren binären Suchbaum nacheinander die Elemente $2, 1, 3, 4, 5, 6, 7$ (in dieser Reihenfolge) ein. Zeichnen Sie den Baum nach jeder Einfüge-Operation.

b) Machen Sie nun das gleiche für einen AVL-Baum.

Aufgabe 6.2 Schreiben Sie eine Prozedur Interval(k_1, k_2), die in einem binären Suchbaum alle Elemente ausgibt, deren Schlüssel zwischen k_1 und k_2 liegen (k_1 und k_2 jeweils eingeschlossen). Sie dürfen annehmen, dass alle Schlüssel paarweise verschieden sind. Beweisen Sie die Korrektheit Ihrer Prozedur und analysieren Sie deren Laufzeit. Die Laufzeit sollte in der Höhe des Baumes und der Länge des ausgegebenen Intervals gemessen werden.

Aufgabe 6.3 Wir betrachten den binären Suchbaum T, den man erhält, wenn man in den leeren Baum die Pivotelemente in der Reihenfolge ihres Auftretens im Quicksort-Algorithmus einfügt.

Genauer: Gegeben sei ein Array $a[1..n]$ von n Elementen. Wir betrachten die Pivotelemente, die Quicksort beim Sortieren dieses Arrays auswählt. Den Quicksort-Aufruf für das komplette Array bezeichnen wir mit dem leeren String, ε. Ist $x \in \{0, 1\}^*$ die Bezeichnung für einen Quicksort-Aufruf, der seinerseits rekursiv Quicksort auf der linken und/oder rechten Hälfte des Arrays aufruft, dann sei $x0$ die Bezeichnung für den Aufruf auf der linken Hälfte und $x1$ die für den auf der rechten Hälfte. Mit p_x bezeichnen wir das von Quicksort-Aufruf x gewählte Pivot-Element. Sei T der binäre Suchbaum, den man erhält, wenn man die Pivotelemente in der Reihenfolge $p_{x_1}, \dots, p_{x_r}$ in den leeren Baum einfügt, wobei $x_1, \dots, x_r$ die Bezeichnungen aller Quicksort-Aufrufe in lexikographisch sortierter Reihenfolge sind.

Zeigen Sie:

a) Wenn T lineare Tiefe in n hat, dann hat Quicksort quadratische Laufzeit in n.

b) Wenn T logarithmische Tiefe in n hat, dann hat Quicksort Laufzeit $O(n \log n)$.