TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Fachbereich Informatik
AG Algorithmen und Komplexität

Ausgabedatum: 11.12.2014
Version: 2014-12-10 16:42

7. Übungsblatt für alle Tracks zur Vorlesung Entwurf und Analyse von Algorithmen, WS 14/15

Abgabe: *Bis Freitag, 19.12.2014, 12:00 Uhr, Kasten im Treppenhaus 48-6.*

Basisaufgaben

B7.1: Aufbauen eines AVL-Baums

3 Punkte

Konstruieren Sie aus der Schlüsselfolge $[91, 11, 49, 85, 56, 77, 32]$ den zugehörigen AVL-Baum. Starten Sie also mit dem leeren Baum, in den Sie den Schlüssel 91 einfügen. Fügen Sie in den resultierenden AVL-Baum den Schlüssel 11 ein, und so weiter.

Stellen Sie jeden so entstehenden Baum einzeln graphisch dar, wobei an jedem Knoten sein Balancegrad notiert werden soll. Wird nach dem Einfügen eine Rotation notwendig, so geben Sie an, welcher Typ von Rotation an welchem Knoten angewendet werden muss; der aus der Rotation resultierende Baum soll dann als neue Graphik dargestellt werden.

Geben Sie auch den aus der Schlüsselfolge $[91, 11, 49, 85, 56, 32, 77]$ resultierenden AVL-Baum an. Was fällt beim Vergleich der beiden Bäume auf, insbesondere in Hinblick auf die Laufzeit von Suchanfragen?

B7.2: Einfügen in Suchbäume

3 Punkte

Stellen Sie für die Folge

$$3, 6, 16, 17, 30, 31, 46, 60, 68, 69,\ 76, 78, 87, 95, 125$$

jeweils das Ergebnis dar, wenn die Elemente von links nach rechts in die anfänglich leere Datenstruktur eingefügt werden; und zwar für:

a) B-Baum der Ordnung $m = 4$

b) Digitaler Suchbaum

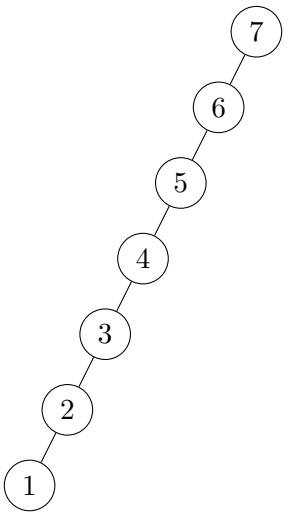
c) Trie

Hinweis: Für die Aufgabenteile b) und c) wandeln Sie die Elemente zunächst in 7-Bit Binärzahlen um.

B7.3: Beitrag einer Splay-Operation zu amortisierten Kosten

3 Punkte

Bestimmen Sie die exakten amortisierten Kosten $\chi_{\mathsf{op}}(T)$ der Operation $\mathsf{op} = \mathsf{splay}(1, T)$ für den Splay-Tree T:



Dabei sei das Potential wie im Buch auf Seite 148 beschrieben: Wir wählen $W(v)$ als Größe des Teilbaums T_v mit Wurzel v, also die Anzahl Nachfolger von v in T (inkl. v selbst). Dann sei der Rang eines Knoten $r(v) = \mathrm{ld}(W(v))$ und das Potential des Baumes schließlich $\phi(T) = \sum_{v \in V} r(v)$.

Ebenfalls wie im Buch nehmen wir an, dass eine Zig- bzw. Zag-Operation (tatsächliche) Kosten 1 verursacht und Zig-Zig/Zig-Zag/Zag-Zig/Zag-Zag jeweils Kosten 2.

B7.4: Datenstruktur für Paare I

3 Punkte

Entwerfen Sie eine Datenstruktur, die Paare aus $\mathbb{N} \times \mathbb{N}$ so speichert, dass ein beliebiges Element in $\mathcal{O}(\log n)$ Zeit gefunden bzw. eingefügt werden kann, wenn n Elemente bereits enthalten sind.

Begründen Sie Korrektheit und Effizienz Ihres Entwurfs.

Aufbauaufgaben

A7.1: Rot-Schwarz-Bäume

2 + 3 + 2 + 4 Punkte

Ein *Rot-Schwarz-Baum* ist ein binärer Suchbaum mit den folgenden Eigenschaften:

(1) Jeder Knoten ist entweder rot oder schwarz.

(2) Die Wurzel ist schwarz.

(3) Ist ein Knoten rot, so sind alle seine Kinder schwarz.

(4) Sei v ein beliebiger Knoten und T_v der Teilbaum, dessen Wurzel v ist. Dann haben alle (einfachen) Pfade von v zu fiktiven Blättern von T_v die gleiche Anzahl an schwarzen Knoten.

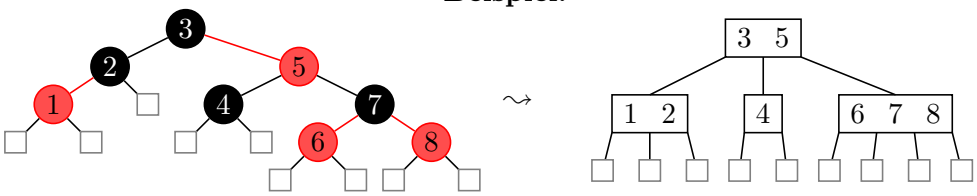
In einem Rot-Schwarz-Baum müssen alle fiktiven Blätter die selbe (fiktive) Farbe haben; per Konvention legen wir fest, dass alle fiktiven Blätter schwarz sind.

Rot-Schwarz-Bäume sind wohl die in der Praxis verbreitetste Variante von binären Suchbäumen; so basiert `java.util.TreeMap` aus der Java Runtime Library ebenso auf Rot-Schwarz-Bäumen wie `std::map` aus der C++ Standard Template Library.

a) Zeigen oder widerlegen Sie: Es gibt $n_0 \in \mathbb{N}$, sodass alle Rot-Schwarz-Bäume mit $n \geq n_0$ Knoten höhenbalanciert sind.

b) Sei ein Rot-Schwarz-Baum T gegeben. Wir konstruieren aus T den zugehörigen „Meta-Baum“ $\tau = \tau(T)$ wie folgt: Für die schwarze Wurzel s von T erzeugen wir die „Meta-Wurzel“ σ in τ, die außer dem Schlüssel aus s auch die Schlüssel aller roten (direkten) Kinder von s enthält. Die Teilbäume von s bzw. seinen roten Kindern transformieren wir rekursiv mit dem gleichen Vorgehen und hängen die „Meta-Teilbäume“ als Kinder von σ ein. Dabei erhalten wir die Reihenfolge der Teilbäume.

Beispiel:



Zeigen Sie, dass τ für jeden gültigen Rot-Schwarz-Baum T ein zulässiger B-Baum der Ordnung $m = 4$ ist. Begründen Sie dazu insbesondere auch, dass oben beschriebene Transformation wohldefiniert ist.

c) Zeigen oder widerlegen Sie: in Rot-Schwarz-Bäumen ist die Suchzeit – erfolgreich wie -los – in $\mathcal{O}(\log n)$, n die Knotenzahl.

d) Entwerfen Sie die Einfügeoperation für Rot-Schwarz-Bäume. Skizzieren Sie dafür die anfallenden Operationen graphisch in allgemeiner Art (vgl. etwa die Beschreibung der Einfügeoperation auf AVL-Bäumen) und geben Sie Pseudocode für die gesamte Operation an. Die Laufzeit sollte in $\mathcal{O}(\log n)$ liegen.

Begründen Sie die Korrektheit Ihres Entwurfs und dass die Laufzeitschranke eingehalten wird.

A7.2: Datenstruktur für Paare II

2 + 2 Punkte

Wir betrachten nochmal das Szenario aus B7.4.

a) Beschreiben Sie, wie man aus Ihrer Datenstruktur aus B7.4 – evtl. mit kleinen Modifikationen, die die in Aufgabe B7.4 geforderten Eigenschaften nicht zerstören – in Zeit $\mathcal{O}(m + \log n)$ eine Liste aller m Elemente $(k, \_)$ in beliebiger Reihenfolge für festes k erzeugen kann.

Begründen Sie Ihre Behauptungen.

b) Nehmen Sie an, dass auf einer festen Instanz eines Wörterbuchs eine (lange) Folge von Suchen nach den Schlüsseln $(k_0, l_0), (k_1, l_1), \ldots$ ausgeführt werden soll. Die Elemente sind nicht – wie üblicherweise angenommen – uniform gezogen, sondern verhalten sich lokal in der ersten Komponente, oder formal

$$\Pr[k_i = k_{i+1}] = 1 - \varepsilon$$

für ein $\varepsilon \in (0, 1)$, wobei wir kleine ε, also das Szenario $\varepsilon \to 0$ untersuchen wollen. Sie dürfen außerdem annehmen, dass es sowohl „genügend viele“ Schlüssel also auch „genügend viele“ verschiedene erste bzw. zweite Komponenten gibt.

Ermöglicht es Ihre Datenstruktur aus B7.4 bzw. a) – evtl. wieder mit kleinen, den allgemeinen Fall nicht störenden Änderungen – aus diesem Wissen Kapital zu schlagen und die mittlere Suchzeit zu verbessern?

Begründen Sie Ihre Behauptung.