INSTITUT FÜR INFORMATIK
KOMPLEXITÄT UND KRYPTOGRAFIE
PROF. DR. JOHANNES KÖBLER

SS 2008
11. JUNI 2008

Theoretische Informatik 3

5. Übung

Abgabe der schriftlichen Lösungen bis zum 2. Juli 2008

Bitte kreuzen Sie die mündlichen Aufgaben zukünftig nur noch in der Gruppe an, in die Sie sich unter Goya eingetragen haben.

Aufgabe 30 Implementieren Sie folgende Prozeduren für einen Suchbaum B. [*4 Punkte*]

a) `Sort`(B) gibt die in B gespeicherten Schlüssel in sortierter Reihenfolge aus.

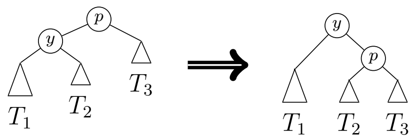
b) `Height`(B) gibt die Höhe von B zurück.

c) `AverageLeafDepth`(B) gibt die durchschnittliche Tiefe der Blätter in B zurück.

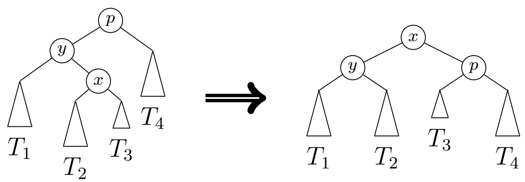
d) `AverageNodeHeight`(B) gibt die durchschnittliche Höhe der Knoten in B zurück.
(*4 Punkte; Teilaufgaben* a) *bis* c) *sind mündlich*)

Aufgabe 31 [*mündlich*]

a) Implementieren Sie eine Prozedur `RightRotate`(y), die den Teilbaum $T(p)$ auf der linken Seite in den Teilbaum $T(y)$ auf der rechten Seite überführt.



b) Implementieren Sie eine Prozedur `LeftRightRotate`(y), die den Teilbaum $T(p)$ auf der linken Seite in den Teilbaum $T(x)$ auf der rechten Seite überführt.



c) Implementieren Sie die in der Vorlesung beschriebene rekursive Prozedur `AVL-Check-Insertion`(y).

Aufgabe 32 Gegeben sei die Einfügesequenz $5, 1, 6, 2, 4, 3$. [*mündlich*]

a) Geben Sie den Suchbaum S an, den die Prozedur `Insert` für binäre Suchbäume bei dieser Einfügesequenz erzeugt.

b) Wie groß ist die Wahrscheinlichkeit, dass `Insert` bei Eingabe einer zufälligen Permutation auf der Schlüsselmenge $\{1, 2, 3, 4, 5, 6\}$ diesen Suchbaum erzeugt?

c) Geben Sie den AVL-Baum T an, den die Prozedur `AVL-Insert` bei dieser Einfügesequenz erzeugt.

d) Wie groß ist die Wahrscheinlichkeit, dass `AVL-Insert` bei Eingabe einer zufälligen Permutation auf der Schlüsselmenge $\{1, 2, 3, 4, 5, 6\}$ diesen AVL-Baum erzeugt?

Aufgabe 33 [*mündlich*]

Implementieren Sie folgende Prozeduren für (un)gerichtete Graphen $G = (V, E)$ mit fester Knotenmenge $V = \{1, \ldots, n\}$, falls G als Feld von Zeigern auf (doppelt verkettete) Adjazenzlisten gespeichert wird.

a) `Init`(G) initialisiert G als den leeren Graphen E_n.

b) `InsertEdge`(u, v) fügt der Kantenmenge E eine neue Kante $\{u, v\}$ bzw. (u, v) hinzu und gibt einen Zeiger e auf den Eintrag dieser Kante in der Adjazenzliste von u zurück.

c) `DeleteEdge`(e) entfernt die Kante, auf die e zeigt.

d) `Edge`(u, v) testet, ob die Kante $\{u, v\}$ bzw. (u, v) in E ist.

Geben Sie entsprechende Prozeduren auch für den Fall an, dass G in einer Adjazenzmatrix gespeichert wird. Geben Sie jeweils asymptotische Schranken für die Laufzeit der Prozeduren an.

Aufgabe 34 [*6 Punkte*]

Sei $G = (V, E)$ ein Digraph mit $V = \{1, \ldots, n\}$. Eine Permutation t auf V heißt *topologische Sortierung* von G, falls für jede Kante $(u, v) \in E$ gilt: $t(u) < t(v)$.

a) Zeigen Sie, dass für G genau dann eine topologische Sortierung existiert, wenn G kreisfrei ist. (*mündlich*)

b) Geben Sie eine Prozedur `TopSort` an, die für einen als Feld von Adjazenzlisten gespeicherten Graphen G in Linearzeit $O(|V|+|E|)$ testet, ob G kreisfrei ist und gegebenenfalls die Knoten von G in topologisch sortierter Reihenfolge ausgibt. (*2 Punkte*)
Hinweis: Führen Sie auf $G^T = (V, E^T)$ eine Tiefensuche aus und geben Sie die Knoten in der Reihenfolge aus, in der sie zum letzten Mal besucht werden.

c) Beweisen Sie sowohl die lineare Laufzeitschranke als auch die Korrektheit Ihrer Prozedur. (*4 Punkte*)