RWTH Aachen **SoSe 2007**
Lehrgebiet Theoretische Informatik Blatt 3
Kneis—Mölle—Richter—Rossmanith 23.4.2007

Übungen zur Vorlesung Datenstrukturen und Algorithmen

T3

Verwenden Sie LuFGTI-Schnipsel, um eine zufällige Permutation der Zahlen $\{1, \ldots, 10\}$ zu erstellen. Dies sollte jeder Teilnehmer und jede Teilnehmerin unabhängig voneinander durchführen. Fügen Sie nun die Zahlen in dieser Reihenfolge in einen leeren Suchbaum ein. Bestimmen Sie die Höhe des entstehenden Suchbaums. Anschließend soll die *durchschnittliche* Höhe aller Bäume in der ganzen Tutorgruppe bestimmt werden.

Lösungsvorschlag:

Gehen Sie mit einer Schere in eine Übungsgruppe.

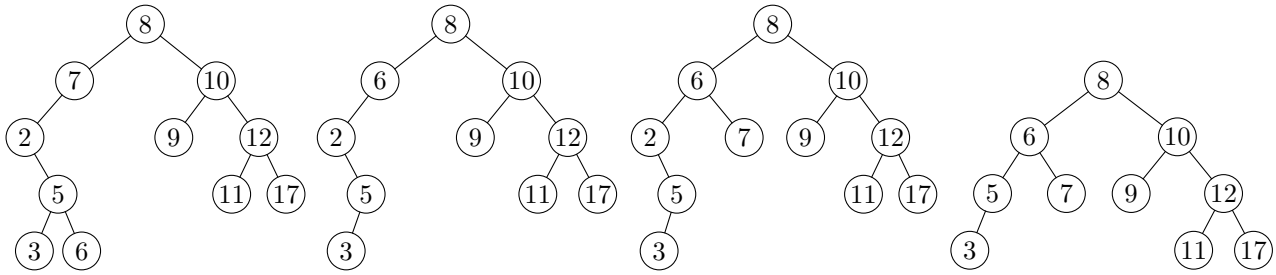
T4

In einen anfangs leeren, binären Suchbaum werden die Elemente 8, 7, 2, 10, 12, 11, 9, 17, 5, und 3 in dieser Reihenfolge eingefügt. Dann wird die 7 gelöscht und anschließend wieder eingefügt. Schließlich wird noch die 2 gelöscht.

Wie sieht der Baum am Ende aus? Verwenden Sie beim Einfügen und Löschen die Methoden der Vorlesung.

Lösungsvorschlag:

Die wichtigsten der hierbei entstehenden Bäume sind:



T5

Bei wievielen Einfügereihenfolgen n verschiedener Elemente in einen leeren Suchbaum entsteht ein Baum mit Höhe n?

Wie groß ist die Wahrscheinlichkeit, daß eine zufällige Einfügereihenfolge zu solch einem entarteten Baume führt?

Lösungsvorschlag:

Ein Baum hat genau dann maximale Höhe, wenn jeder innere Knoten nur ein Kind hat. Dieses kann ein rechtes oder linkes Kind sein, was zwei Möglichkeiten ergibt. Es gibt $n-1$ innere Knoten, also insgesamt 2^{n-1} Möglichkeiten. Für jede dieser „Baumformen“ gibt es genau eine Einfügungsreihenfolge: Die Wurzel ist eindeutig bestimmt, und der Rest folgt mit Induktion.

Da es $n!$ Einfügereihenfolgen gibt, beträgt die Wahrscheinlichkeit für Bäume maximaler Höhe genau $2^{n-1}/n!$.

T6

Jemand sagt, folgende Datenstruktur bewältige m Operationen in $O((n+m)(n+\log m))$ Schritten. Als Operationen seien Einfügen, Suchen und Löschen gestattet, und n sei die anfängliche Größe des Suchbaums.

Die Datenstruktur verwendet einen binären Suchbaum, zählt aber die Operationen mit. Sobald diese Zahl die Hälfte der aktuellen Größe des Suchbaums überschreitet, wird der Suchbaum in linearer Zeit neu aufgebaut, so daß er nur noch logarithmische Höhe hat.

Stimmt seine Behauptung?

Lösungsvorschlag:

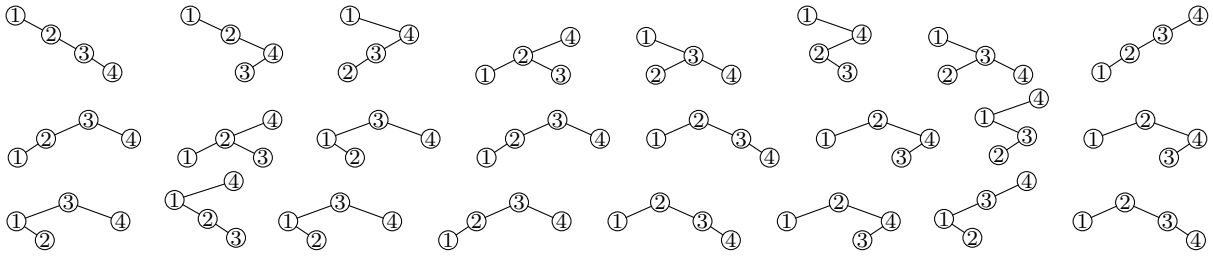
Leider hat er nicht recht. Werden die Zahlen $1, \ldots, m$ in einen anfangs leeren Baum eingefügt, dann betrachten wir die erste Reorganisation nachdem der Baum die Größe $m/4$ überschritten hat. Jetzt gibt es mindestens $m/8$ Einfügeoperationen *ohne* Reorganisation. Die zweite Hälfte davon, also $m/16$ viele, fügen ein Element an das Ende eines Baumes, der mindestens Höhe $m/16$ hat. So ein Einfügen benötigt also $\Omega(m)$ Zeit und es gibt ja $\Omega(m)$ viele solche bösen Einfügeoperationen. Die Zeit für diese beträgt bereits $\Omega(m^2)$.

H6 (5 Punkte)

Vier verschiedene Elemente in zufälliger Reihenfolge werden in einen anfangs leeren, binären Suchbaum eingefügt. Was ist die genaue mittlere Höhe der entstehenden Bäume?

Lösungsvorschlag:

Die 24 Suchbäume sehen so aus:



Wir sehen, daß 16 Bäume die Höhe drei und acht Bäume die Höhe vier haben. Im Durchschnitt ergibt das genau 10/3.

H7 (10 Punkte)

Konstruieren Sie einen optimalen Suchbaum, der die Schlüssel 1, 2, 3 und 4 enthält. Die jeweiligen Zugriffswahrscheinlichkeiten mögen 1/3, 1/4, 1/4 und 1/6 betragen.

Lösungsvorschlag:
Es ergibt sich folgender optimaler Suchbaum: