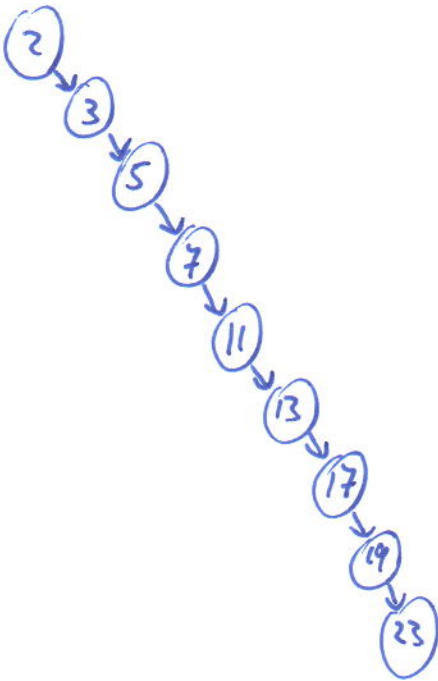


4.6 AVL-Bäume

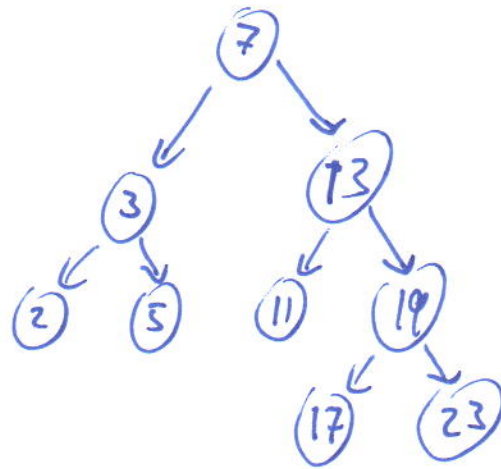
Entscheidend bei binären Suchbäumen: Höhe!

(Einfügen und Entfernen jeweils in $O(h)$)

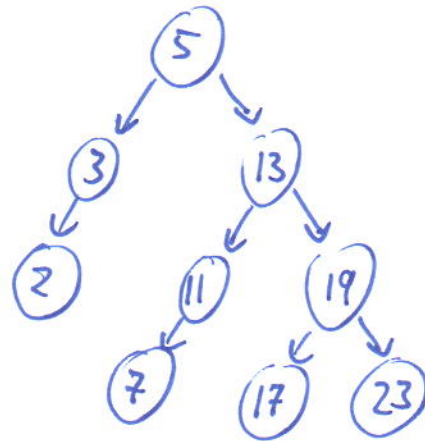
Schlecht:



Gut:



Auch gut:



Idee!

Gestalte Baum „balanciert“, d.h. linken und rechten Teilbaum gleich groß.

Problem!

Nicht immer möglich!

Idee 2: Betrachte „annähernd balanciert“.

Problem 2: Gewicht ist keine lokale Eigenschaft bei lokalen Eigenschaften - eher global...
(Wie setzt man das schnell und lokal um ?!)

Idee 3: Wichtig ist eigentlich gar nicht das Gewicht, sondern die Tiefe/Höhe!

Das verfolgt

DEFINITION 4.7 (AVL-Baum, nach Adelson-Velskij und Landis, 1962)

- (1) Ein binärer Suchbaum ist höhenbalanciert, wenn sich für jeden inneren Knoten v die Höhe der beiden Kinder von v um höchstens 1 unterscheidet.
- (2) Ein höhenbalancierter Suchbaum heißt auch AVL-Baum.

Beispiele „gut“ auf vorheriger Seite sind beide höhenbalanciert.

Problem 3: Reicht „höhenbalanciert“, um logarithmische Höhe zu garantieren?

Problem 4: Wie sorgt man dafür, dass die Eigenschaft „höhenbalanciert“ bei INSERT und DELETE erhalten bleibt?

Beobachtung:

Wenn ein Baum höhenbalanciert ist, sind es auch alle seine Teilbäume, d.h. Höhenbalanciertheit ist eine rekursive Eigenschaft!

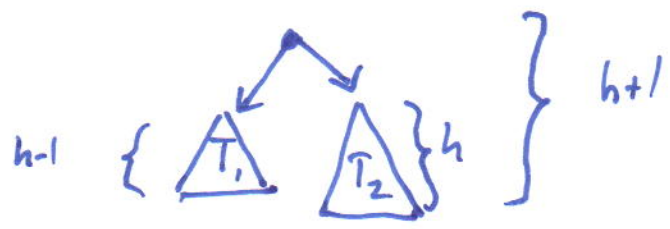
Idee 3:

Wieviele Knoten braucht man mindestens für einen AVL-Baum der Höhe h ?

(D.h. wir untersuchen nicht Höhe in Abhängigkeit von der Knotenzahl sondern Knotenzahl in Abhängigkeit von der Höhe!)

wenn n mindestens exponentiell in h wächst, dann wächst h höchstens logarithmisch in n .

Betrachte allgemein:



Mit $n(1)=1$ und $n(2)=2$ • bzw. \Downarrow
 und $n(h+1) = 1 + n(h) + n(h-1)$
 erhält man

h	$n(h)$
1	1
2	2
3	4
4	7
5	12
6	20
7	33
8	54

(Sehr ähnlich:
 $f(0)=1, f(1)=1, f(h+1) = f(h) + f(h-1)$
 liefert 1, 1, 2, 3, 5, 8, 13, 21, 34
 → Fibonacci-Zahlen!)

Sauber bewiesen:

SATZ 4.8

Die Höhe eines AVL-Baumes mit n Knoten ist $O(\log n)$.

BEWEIS:

Statt einer oberen Schranke für die Höhe eines AVL-Baumes zeigen wir zunächst eine untere Schranke für die Zahl der Knoten eines AVL-Baumes!

Sei $n(h)$ die kleinste Zahl von Knoten eines AVL-Baumes der Höhe h .

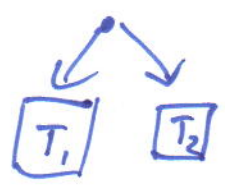
Wir beobachten

$$\begin{aligned} n(0) &= 1 \\ n(1) &= 2 \end{aligned}$$

(ein Knoten erforderlich)
(zwei Knoten erforderlich wegen Höhe)

Noch einmal gilt

$$n(h) = \underset{\substack{\uparrow \\ \text{Wurzelknoten}}}{1} + \underset{\substack{\uparrow \\ \text{Teilbaum}}}{n(h-1)} + \underset{\substack{\uparrow \\ \text{Teilbaum}}}{n(h-2)}$$



Wir zeigen per Induktion:

$$n(h) \geq 2^{\frac{h}{2} - 1}$$

Induktionsanfang:

Die Behauptung gilt für

$$\begin{aligned} h=1 & : n(1) = 1 \geq 2^{-\frac{1}{2}} = 2^{\frac{1}{2}-1} \\ h=2 & : n(2) = 2 \geq 2^0 = 2^{\frac{2}{2}-1} \end{aligned}$$

Induktionsschritt:

Gelte die Behauptung für alle $h \in \{0, \dots, k\}$.

Wir zeigen: Die Behauptung gilt für $h=k+1$!

Betrachte also $n(k+1) \geq 1 + n(k) + n(k-1)$.

Da $n(h)$ monoton wächst (für größere Höhe braucht man mindestens so viele Knoten wie für niedrigere!), gilt

$$n(k) \geq n(k-1)$$

Also ist $n(k+1) \geq 1 + 2 \cdot n(k-1)$
 $\geq 1 + 2 \cdot 2^{\frac{k-1}{2}-1}$
 Ind.annahme \uparrow
 $= 1 + 2^{\frac{k+1}{2}-1}$

Damit gilt die Behauptung auch für $h=k+1$.

Induktionsschluss:

Für alle $h \in \mathbb{N}$ gilt $n(h) \geq 2^{\frac{h}{2}-1}$.

Damit ist auch

$$\log_2(n(h)) \geq \frac{h}{2} - 1$$

oder $h \leq 2(\log_2 n + 1)$

denn $n(h)$ ist die kleinstmögliche Knotenzahl bei Höhe h .

Damit hat ein AVL-Baum mit n Knoten höchstens die Höhe $2(\log_2 n + 1)$, also $O(\log n)$.

□

Es bleibt Problem 4:

Wie bewahrt man Höhenbalanciertheit bei INSERT und DELETE?