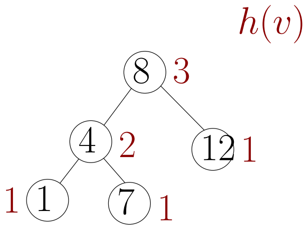
AVL Bäume

- Name AVL von Erfindern Adelson-Velskii und Landis
- Erfunden 1962, älteste Datenstruktur für balancierte Bäume

AVL Baum

Def.: ein binärer Suchbaum T heißt AVL Baum falls sich für jeden Knoten v von T die Höhen der Unterbäume um höchstens 1 unterscheiden

- Wir speichern Höhe $h(v)$ für jeden Knoten v

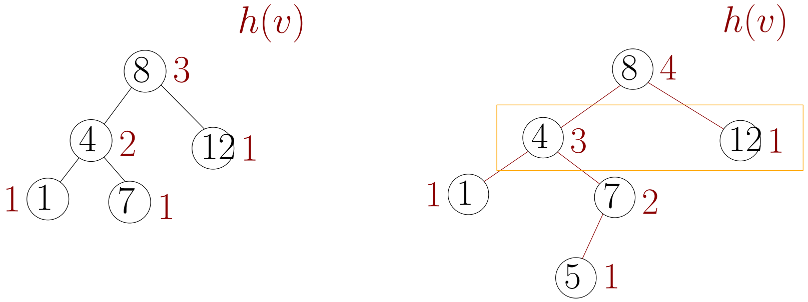


Operationen

- Aufzählen - kann direkt von BST übernommen werden
- Suche - kann direkt von BST übernommen werden
- Einfügen - BST Methode kann zu Problemen führen (AVL Eigenschaft stimmt evtl. nicht mehr)
- Löschen - BST Methode kann zu Problemen führen (AVL Eigenschaft stimmt evtl. nicht mehr)

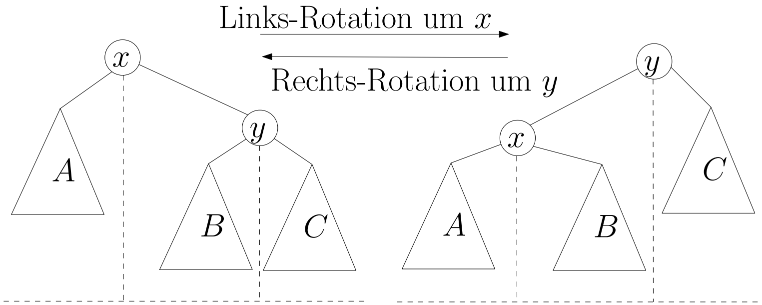
Problem

Durch Operationen Einfügen (im Bsp. von Knoten mit Schlüssel 5) und Löschen wird AVL Eigenschaft verletzt



Rotation

Um Baum wieder zu balancieren, wird er lokal umstrukturiert, so dass die Schlüsselordnung erhalten bleibt:



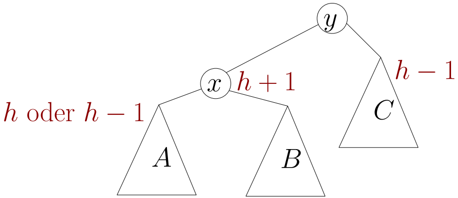
Links-Rotation kann nur auf *x* angewendet werden, wenn *x* ein rechtes Kind hat (symmetrisch für Rechts-Rotation)

Balancieren

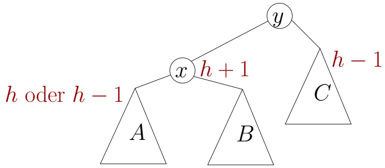
- **Beinahe-AVL-Baum:** AVL-Kriterium bei allen Knoten erfüllt außer bei der Wurzel. Höhenunterschied zwischen linkem und rechtem Teilbaum der Wurzel ist genau 2.
- Rotationen können genutzt werden, um einen beinahe-AVL-Baum zu balancieren.

Balancieren

Annahme: linker Teilbaum größer als rechter Teilbaum (anderer Fall symmetrisch)



Balancieren



Fall 1: A hat Höhe h

$\Rightarrow$ B hat Höhe h oder $h - 1$

$\Rightarrow$ Rechts-Rotation um y ergibt AVL-Baum

Fall 2: A hat Höhe $h - 1$

$\Rightarrow$ B hat Höhe h

$\Rightarrow$
1. Links-Rotation um x
2. Rechts-Rotation um y

(auch Doppelrotation genannt) ergibt AVL-Baum

Balancieren

- Ein beinahe-AVL-Baum kann mit maximal zwei Rotationen balanciert werden
- Die Funktion, die dies erzielt, nennen wir im Folgenden BALANCE(KNOTEN X)

Einfügen

Algorithm 1: AVL_Einfügen(Knoten x von Baum T, Knoten z)

if *(x = TNULL)* **then**
    /*Füge den Knoten als Blatt in den Baum ein*/
    h(z) = 1
end
else if *(Schlüssel(z) $<$ Schlüssel(x))* **then**
    AVL_Einfügen(Left(x), z)
end
else if *(Schlüssel(z) $>$ Schlüssel(x))* **then**
    AVL_Einfügen(Right(x), z)
end
else
    /*Schlüssel schon in T*/
end
h(x) = $1 + \max(h(Left(x)), h(Right(x)))$
Balance(x)

Einfügen

- Einfügen führt zu maximal $O(h)$ Aufrufen von BALANCE
- Aber: Einfügen führt zu maximal einem beinahe-AVL-Baum (d.h. BALANCE hat maximal ein Mal etwas zu tun)
- Damit: maximal zwei Rotationen werden durchgeführt
- Gesamtzeit Einfügen: $O(h)$

Löschen

- Lösche Element wie in BST
- Balanciere Braum von der untersten veränderten Stelle entlang des Pfads zur Wurzel neu
- Maximal h Doppelrotationen in BALANCE möglich
- Gesamtzeit Löschen: $O(h)$

Aufwansabschätzung

Frage: was ist die maximale Höhe eines AVL Baums mit n Knoten?

Verwandte Frage: was ist die minimale Knotenzahl, genannt min_h, eines AVL Baums der Höhe h?

$$\mathrm{min}_0 = 0$$
$$\mathrm{min}_1 = 1$$
$$\mathrm{min}_h = \mathrm{min}_{h-1} + \mathrm{min}_{h-2} + \underbrace{1}_{\text{Wurzel}}$$

Grund: AVL Eigenschaft

Ausflug: Fibonacci-Zahlen F_i

$$F_0 = 0$$
$$F_1 = 1$$
$$F_i = F_{i-1} + F_{i-2}$$

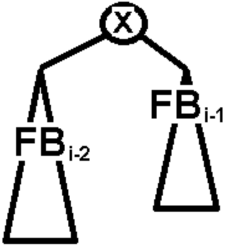
Satz 1: Es ist $F_h = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^h - \left(\frac{1-\sqrt{5}}{2}\right)^h\right)$.

Ausflug: Fibonacci-Bäume FB_i

Fibonacci Baum der Höhe 0 FB_0: leerer Baum

Fibonacci Baum der Höhe 1 FB_1: ein Knoten

Fibonacci Baum der Höhe i FB_i: Baum x, FB_{i-1}, FB_{i-2}



Zusammenhang

- Fibonacci-Bäume haben $\min_h$ Knoten, d.h. sie sind für eine gegebene Baumhöhe die “schiefsten” AVL-Bäume mit minimaler Knotenanzahl.
- **Satz 2:** Es ist $\min_h = F_{h+2} - 1$.
- Beweis: vollständige Induktion.

Zusammenhang

- Für einen AVL-Baum mit n Knoten ist (per Definition) $n \geq \min_h$ (h ist hier unbekannt)
- $\min_h = F_{h+2} - 1$ (Satz 2)
- $F_{h+2} = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^{h+2} - \left(\frac{1-\sqrt{5}}{2} \right)^{h+2} \right)$ (Satz 1)
- "Einsetzen": $n \geq \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^{h+2} - \left(\frac{1-\sqrt{5}}{2} \right)^{h+2} \right) - 1$
- Damit: $h = O(\log n)$

Zusammenfassung AVL-Bäume

Laufzeit der Operationen in einem AVL-Baum mit n Knoten:

Aufzählen	$O(n)$
Suchen	$O(\log(n))$
Einfügen	$O(\log(n))$
Löschen	$O(\log(n))$