

Bäume sind dynamische Datenstrukturen, durch Einfügen/Löschen kann der Baum von ausgeglichen schnell zu entartet wechseln!

⇒ Höhenbalanciert (gut für viel Sachen, wenig rekurrenz)

2-4 Baum: (es gibt auch 8-16 Bäume, etc...)

⇒ Die Daten selbst stehen in den Blättern

⇒ Die Knoten enthalten Zusatzinfos, in den Bsp. immer das Maximum des Teilbaumes aus dem Sohn;

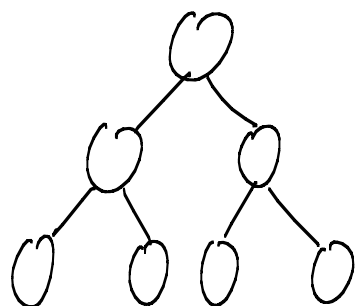
Beobachtungen:

- alle Äste gleich lange
- # Söhne $\rightarrow 2 \leq t \leq 4$

Bsp.:

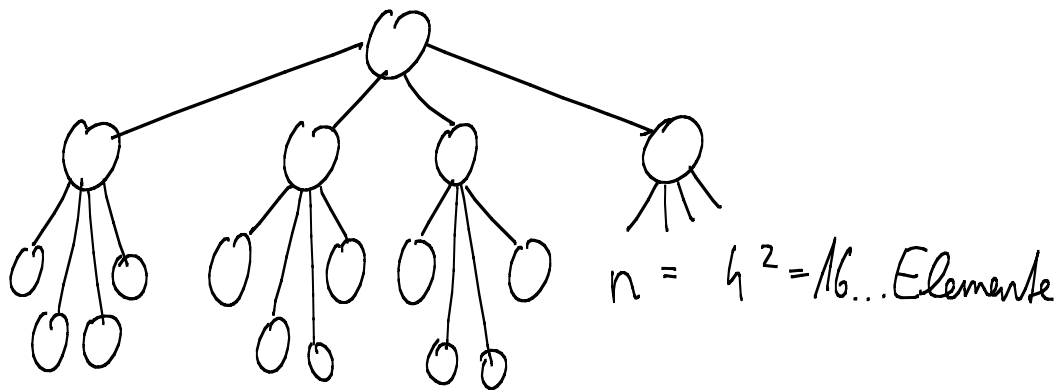
Höhe = 2

alle Knoten haben 2 Söhne



$n = 2^2 = 4 \dots$ Elemente

Bsp.: Höhe = 2
alle Knoten haben 4 Söhne



Beobachtung: allgemein $n = 4^h \dots \# \text{Blätter}$

$$2^h \leq n \leq 4^h \quad h \dots \text{Höhe des Baumes}$$

$$2^h \leq n \leq 2^{2h} \quad | \log$$

$$h \leq \log(n) \leq 2h$$

Höhe: $\frac{\log(n)}{2} \leq h \leq \log(n) \quad / \hat{=} \Theta \text{ Notation}$

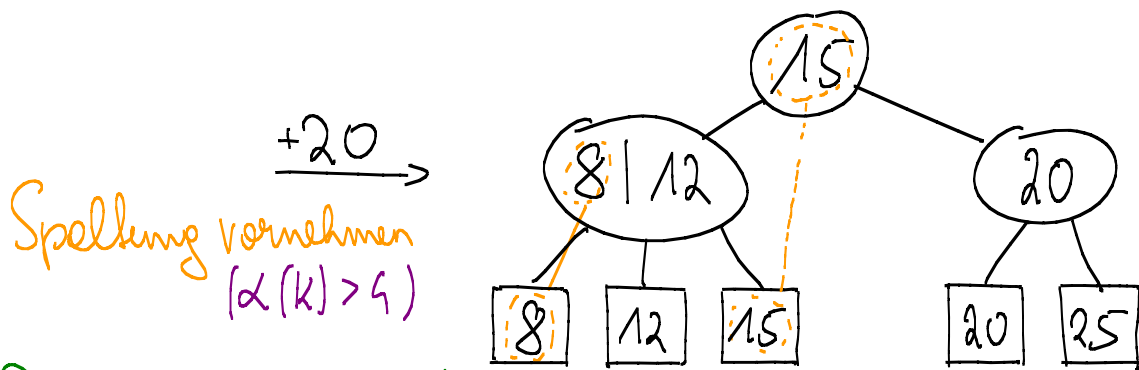
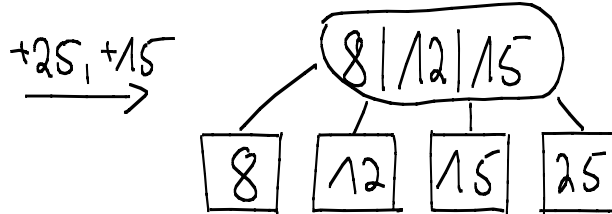
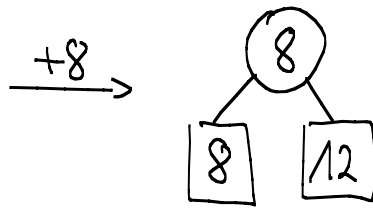
$h = \Theta(\log(n))$ Höhe ist von oben & unten durch $\log(n)$ beschränkt

Wörterbuchproblem: Suchen, löschen, eintragen
von Daten in Datenstrukturen

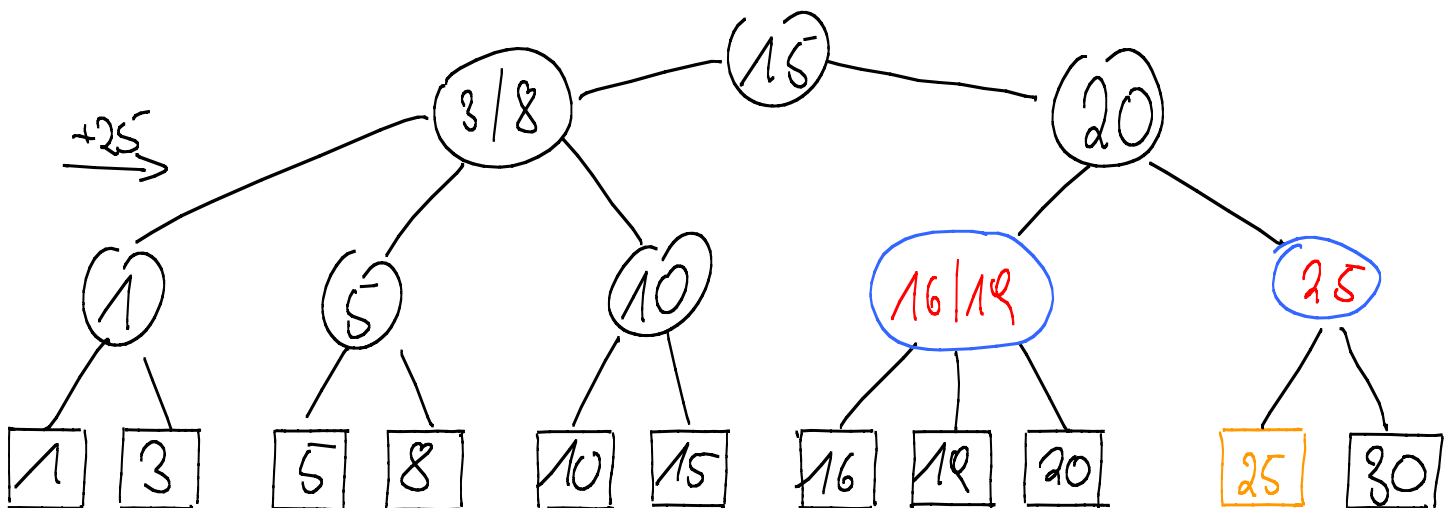
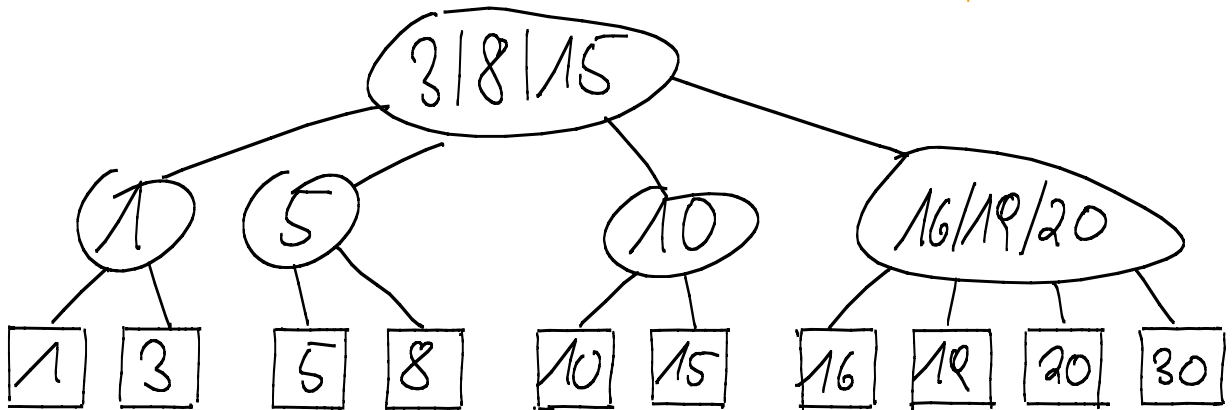
Ein 2-4 Baum muss nach löschen/einfügen reorganisiert werden, um die 2-4 Eigenschaften zu halten.
Dies kostet Laufzeit, die sich jedoch amortisiert

Beispiel:

Leerer Baum $\xrightarrow{+12}$ $\boxed{12}$



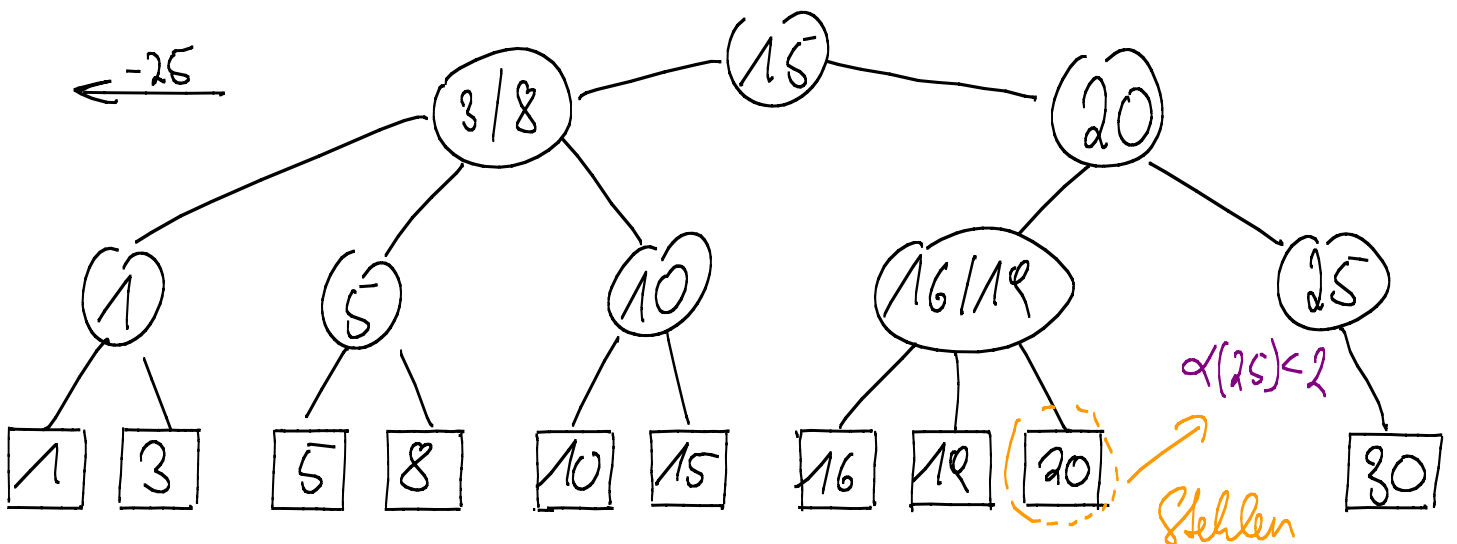
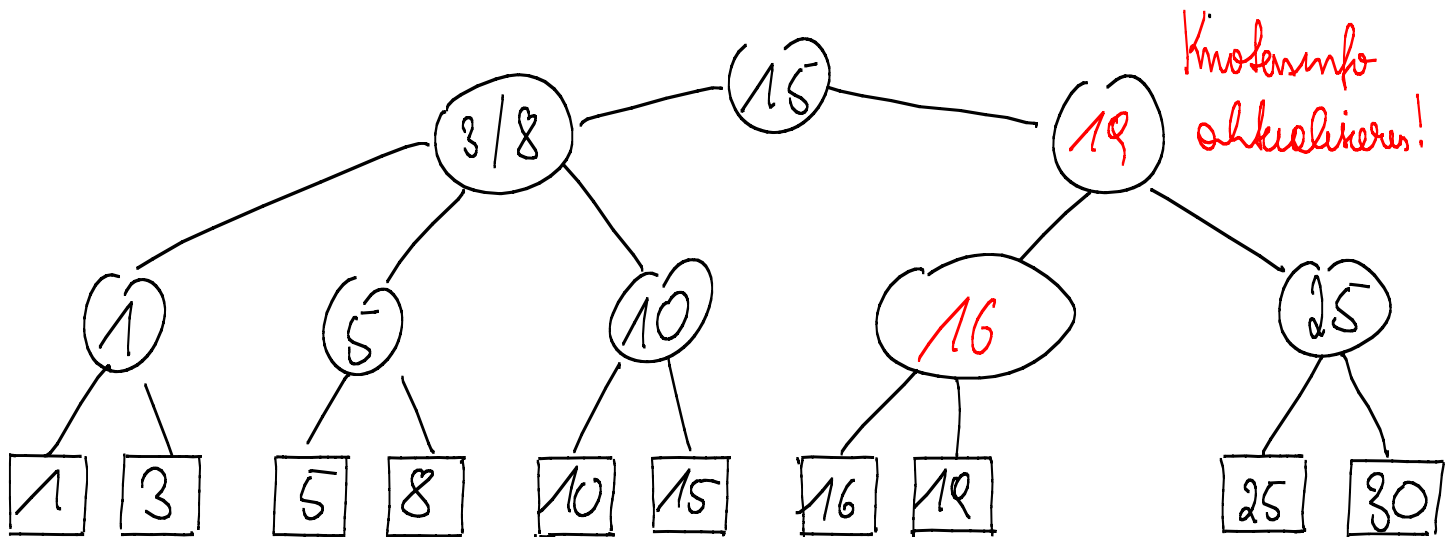
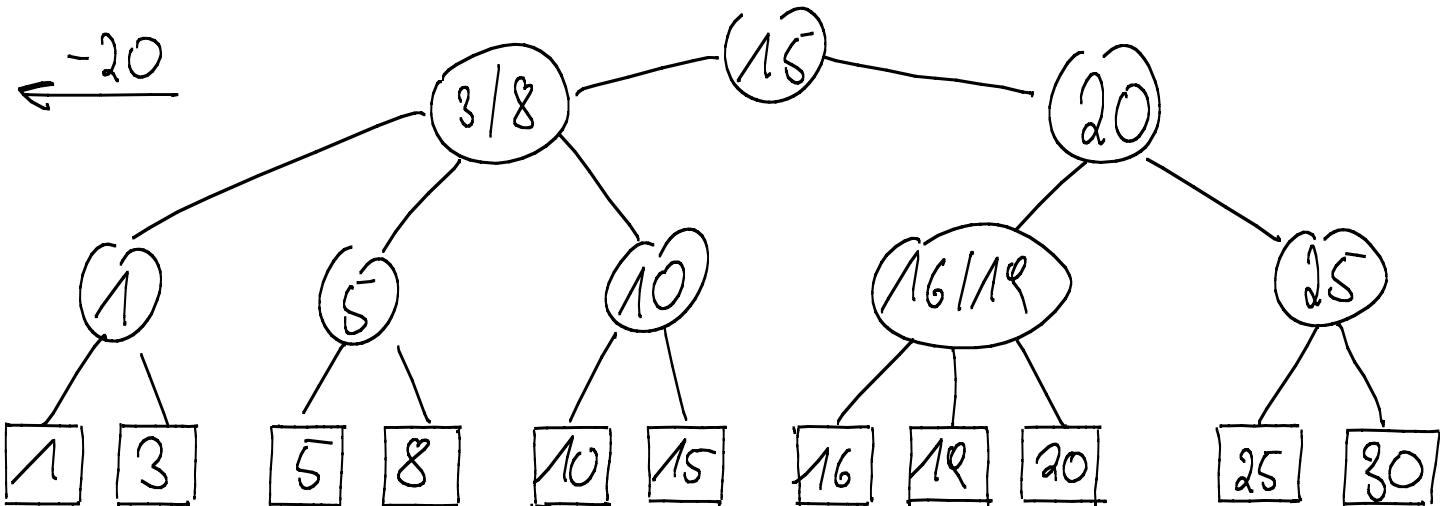
Der linke behauptet 3 Blätter, aber Rechte 1;
Max. des linken Teilbaumes als Zusatzinfo;

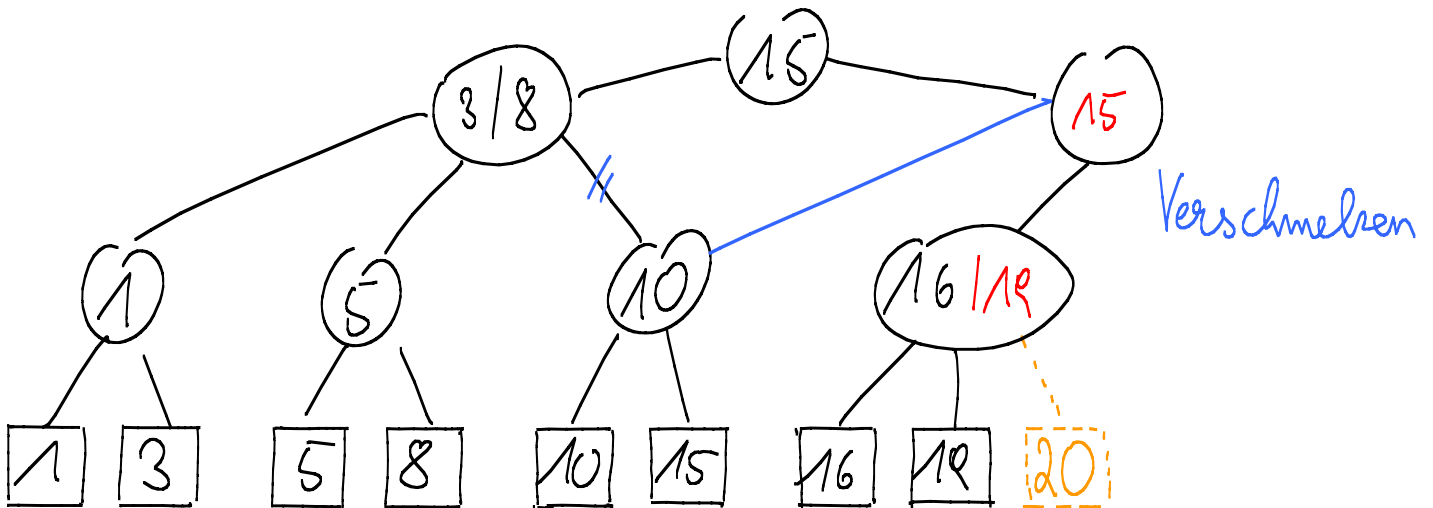
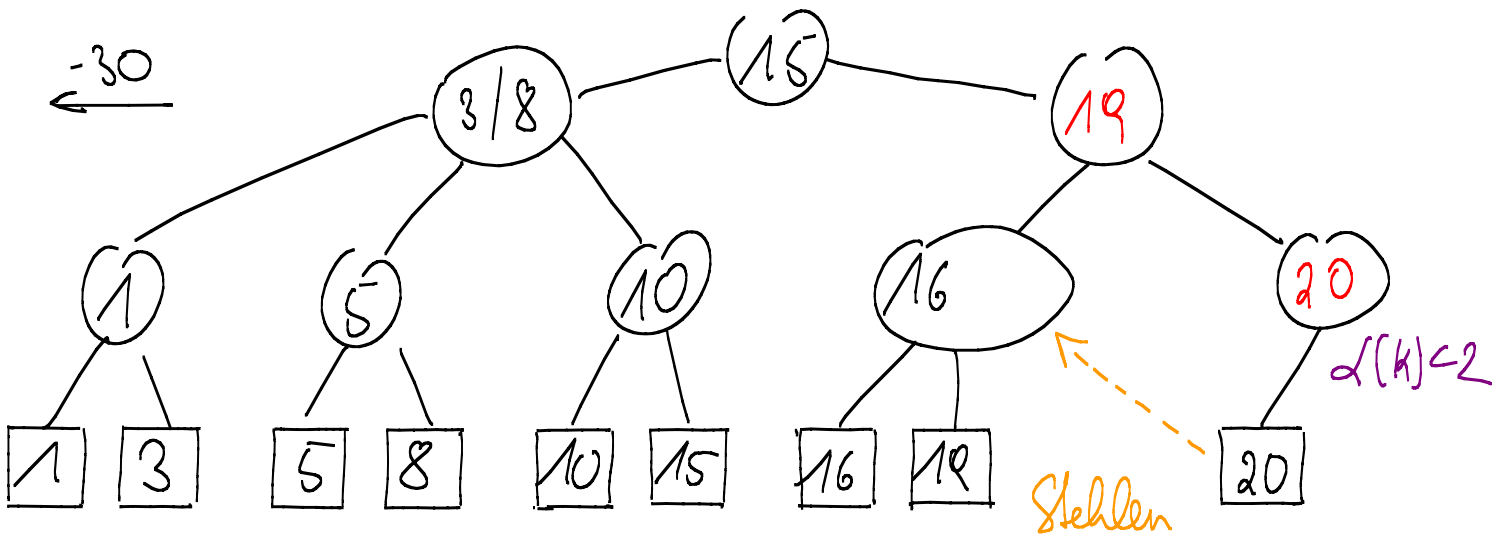
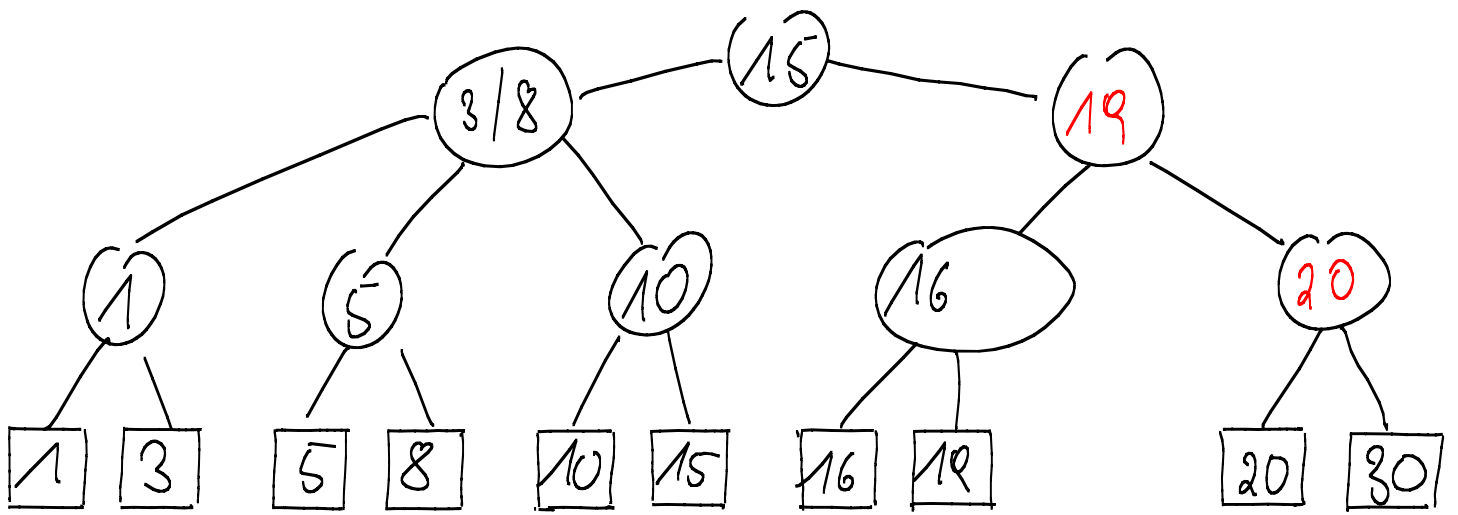


$$T(n) = O(\log(n)) + O(1) \log(n)$$

$$= O(\log(n))$$

Beim Löschen ist die Idee Knoten mit 3 Söhnen zu erhalten;





Speicher : $O(n)$... kein zusätzlicher Speicher notwendig
 Einfügen, löschen, suchen in $O(\log(n))$

Amortisierte Kosten für (2,4) Bäume:

Definition:

$$\text{balance } b_k(\text{Knoten } k) \begin{cases} -1 \\ 0 \\ 1 \\ 0 \\ -1 \end{cases} \xleftarrow{\text{ketgelegt}} \alpha(k) = \begin{cases} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{cases}$$

im Lauf der Operationen
kommen 1, 5 auch auf!:

$$S(T) = \sum S(k) \quad \dots \text{optimal ist } \alpha(k) = 3 \rightarrow b(k) = 1 \text{ für alle Knoten}$$

↳ T... Tree = der gesamte Baum

i ... # Einfügeoperationen

j ... # Löschoptionen

$$m = i + j$$

S ... # Spaltvorgänge

V ... # Verschmelzen

D ... # Stehlen

1. Satz: $D \leq j$... wenn gelöscht wird muss ev.
nicht jedes mal geteilt werden

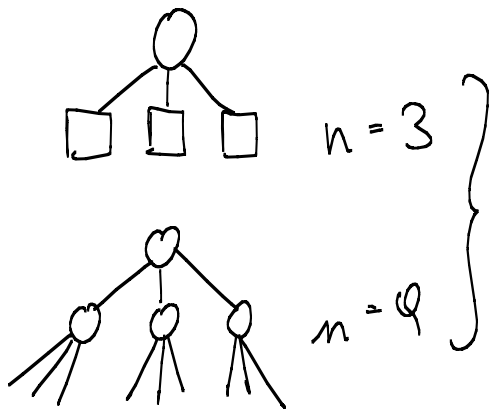
2. Satz: $S + V \leq m + \frac{i - j}{2}$

Beobachtungen:

① $L(T) = \text{begrenzt}$

$$L(T) \geq 0$$

weil alle Knoten $\alpha(k) = 4$ oder $\alpha(k) = 2$



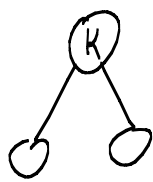
allgemein: $n \cdot \sum_{i=1}^{\log(n)} \left(\frac{1}{3}\right)^i$

$$= n \left[\sum_{i=1}^{\log(n)} \left(\frac{1}{3}\right)^i - 1 \right] < n \left[\sum_{i=0}^{\infty} \left(\frac{1}{3}\right)^i - 1 \right] = \frac{n}{2}$$

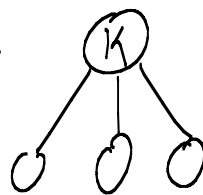
$$\frac{1}{1 - \frac{1}{3}} = \frac{3}{2}$$

① Einfügen & Entfernen

$$b(k) = 0$$



Einfügen



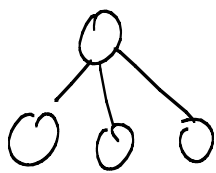
$$b = +1$$

Entfernen

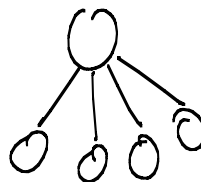


$$b = -1$$

$$b(k) = 1$$

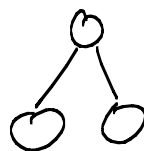


Einfügen



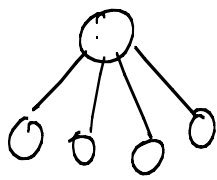
$$b = 0$$

Entfernen

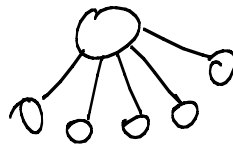


$$b = 0$$

$$b(k) = 0$$

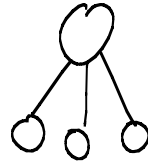


Einfügen →



$$b = -1$$

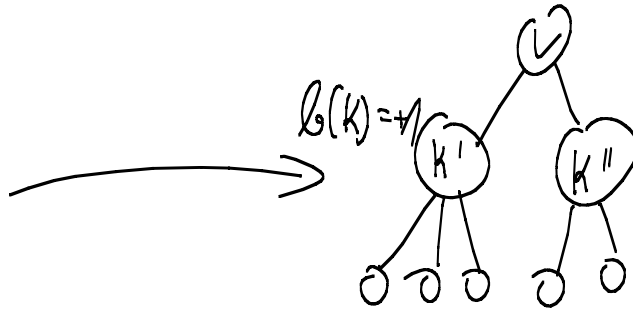
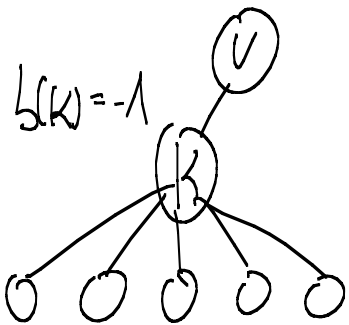
↘ Entfernen



$$b = +1$$

⇒ $\Delta b(k)$ ist ± 1 je nach Operation

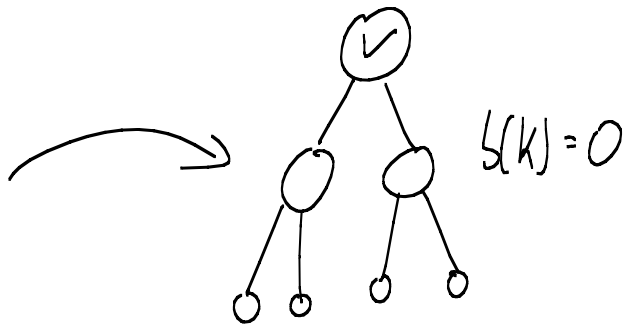
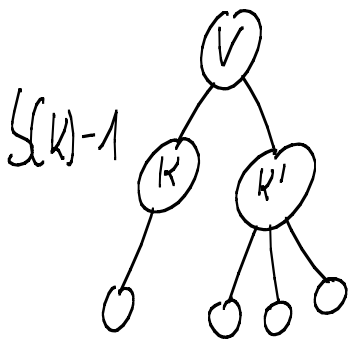
III Spalten



$$b(T') \geq b(T) - 1 - b(k) + b(k') + b(k'')$$

$$\Rightarrow b(T') \geq b(T) + 1$$

IV Stehlen:



$$\Rightarrow b(T') \geq b(T)$$

V Verschmelzen (ähnlich Stehlen)

$$\Rightarrow b(T') \geq b(T) + 1$$

Abrechnung:

leeres T_0
 m Operationen
Abbruch
Einsetzen

$$S(T_0) = 0 \rightarrow S(T_m) \stackrel{\textcircled{I}}{\leq} \frac{i-j}{2} = \frac{m}{2}$$

$$\#_j \prec \#_i \stackrel{\textcircled{II}}{\leq} m$$
$$S+V$$

$$\Rightarrow 0 - m + (S+V) \leq \frac{i-j}{2}$$

$$\text{Sch 1+2: } S+V+D \leq m + \frac{i-j}{2} \cdot j$$
$$= \frac{2i + 2j + i - j + 2j}{2}$$
$$= \frac{3i + 3j}{2}$$
$$= \frac{3}{2} m$$

$$\Rightarrow S+V+D = O(m)$$

Da sich der Baum bei jedem Einfügen/
Löschen etwas ausgleicht;

offensichtlich ist $O(\log n)$ keine scharfe
Grenzsetzung;

Die Laufzeit steigt linear mit den
Einfüge & Löschoptionen