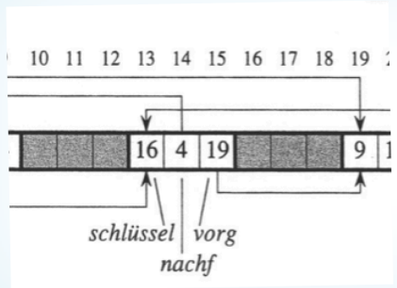
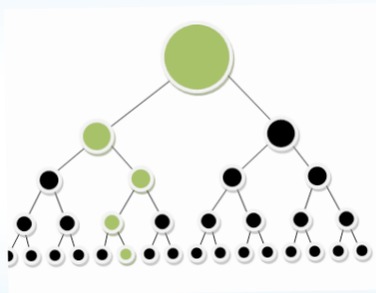





Kapitel 4.6: AVL-Bäume

Algorithmen und Datenstrukturen
WS 2024/25

Prof. Dr. Sándor Fekete

4.5 Binäre Suchbäume

<u>Schnell:</u>

- ***O(log n): logarithmische Zeit***
- ***O(h): Tiefe des Baumes***

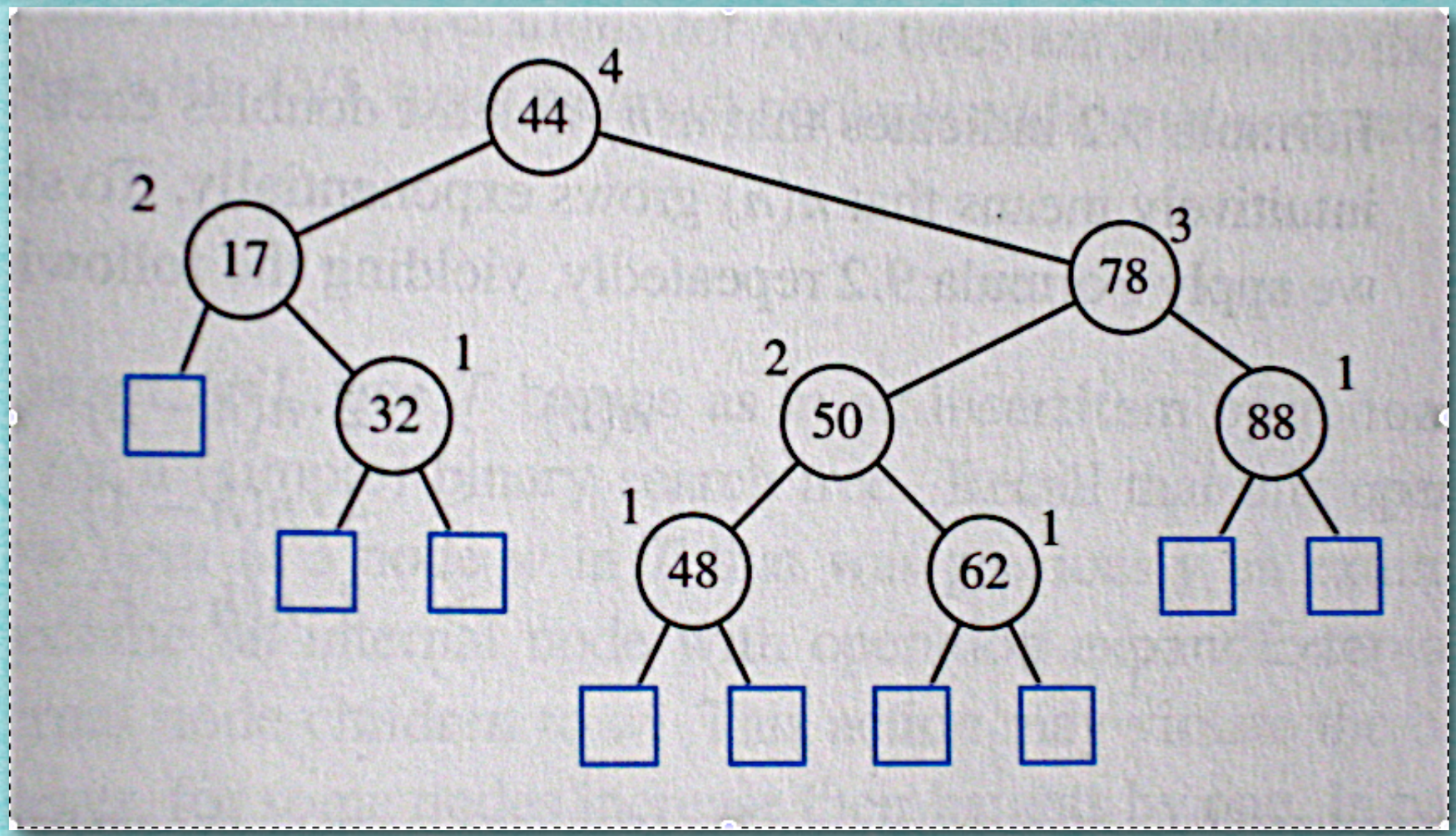
<u>Also: Wie können wir die Tiefe des Baumes auf O(log n) beschränken?</u>

4.6 AVL-Bäume

Definition 4.7 (Nach Adel'son-Vel'skiĭ und Landis, 1962)

(1) Ein binärer Suchbaum ist ***höhenbalanciert***, wenn sich für jeden inneren Konten v die Höhe der beiden Kinder von v um höchstens 1 unterscheidet.

(2) Ein höhenbalancierter Suchbaum heißt auch ***AVL-Baum***.



Ab an die Tafel!

s.fekete@tu-bs.de

Satz 4.8
Ein AVL-Baum mit n Knoten hat höchstens Höhe O(log n).

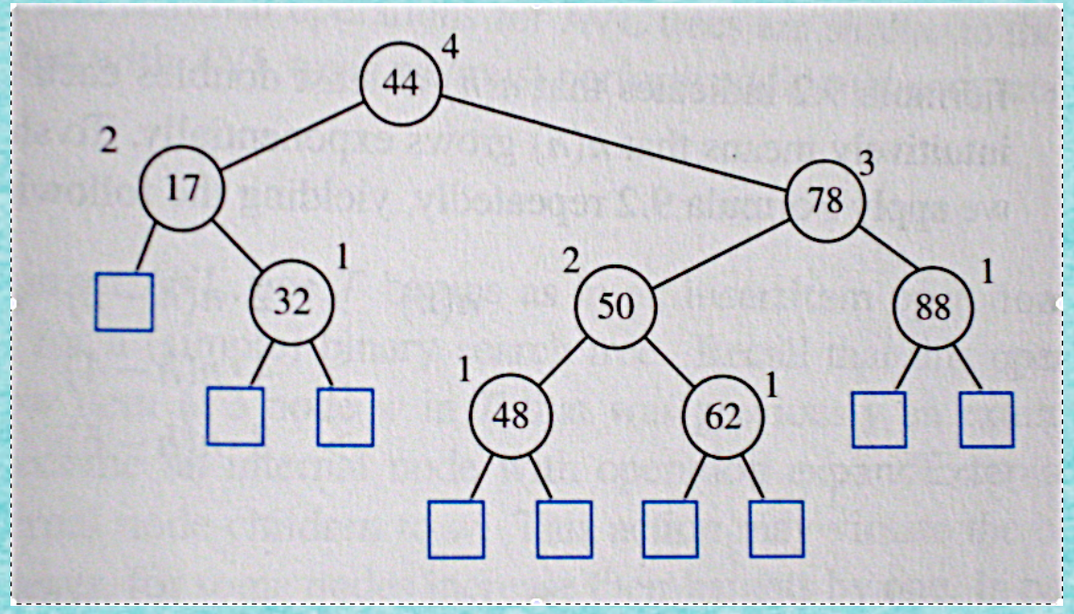
Beweis:
Wie gesehen!

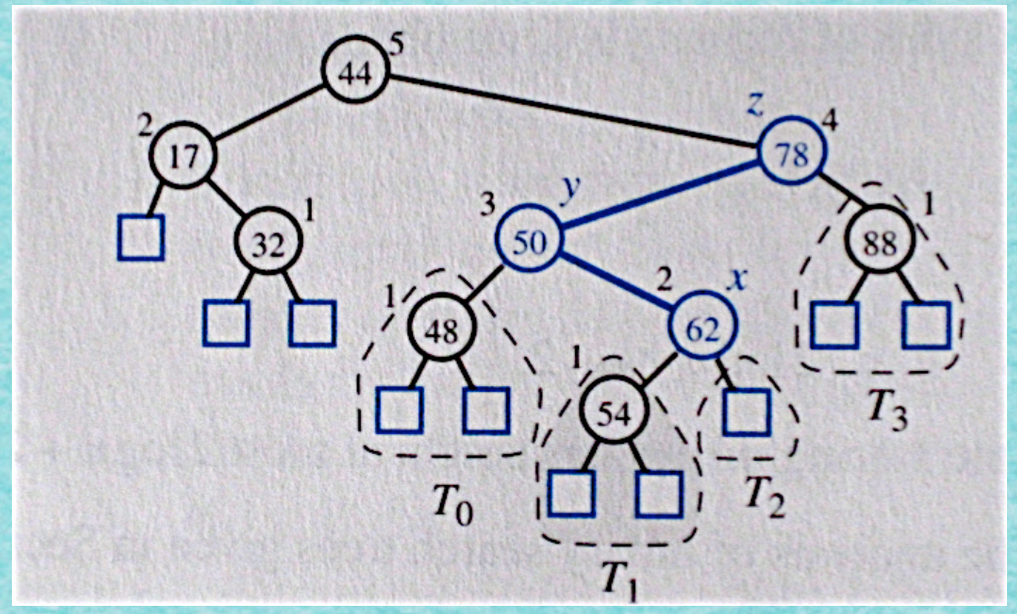
Damit noch offen:
Wie erhält man Höhenbalanciertheit in dynamischen Situationen?

Einfügen („INSERT“)

Aufgabe:

- ***Füge 54 ein!***

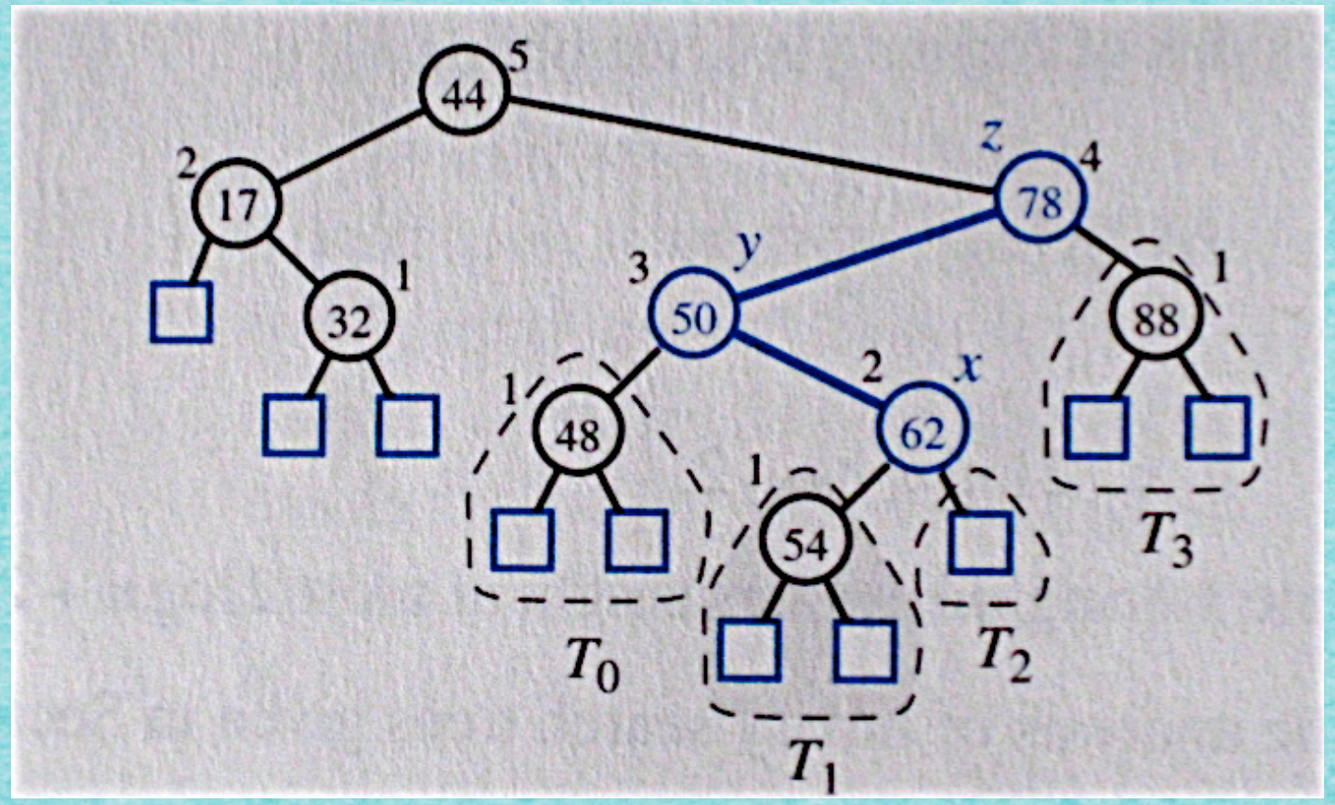




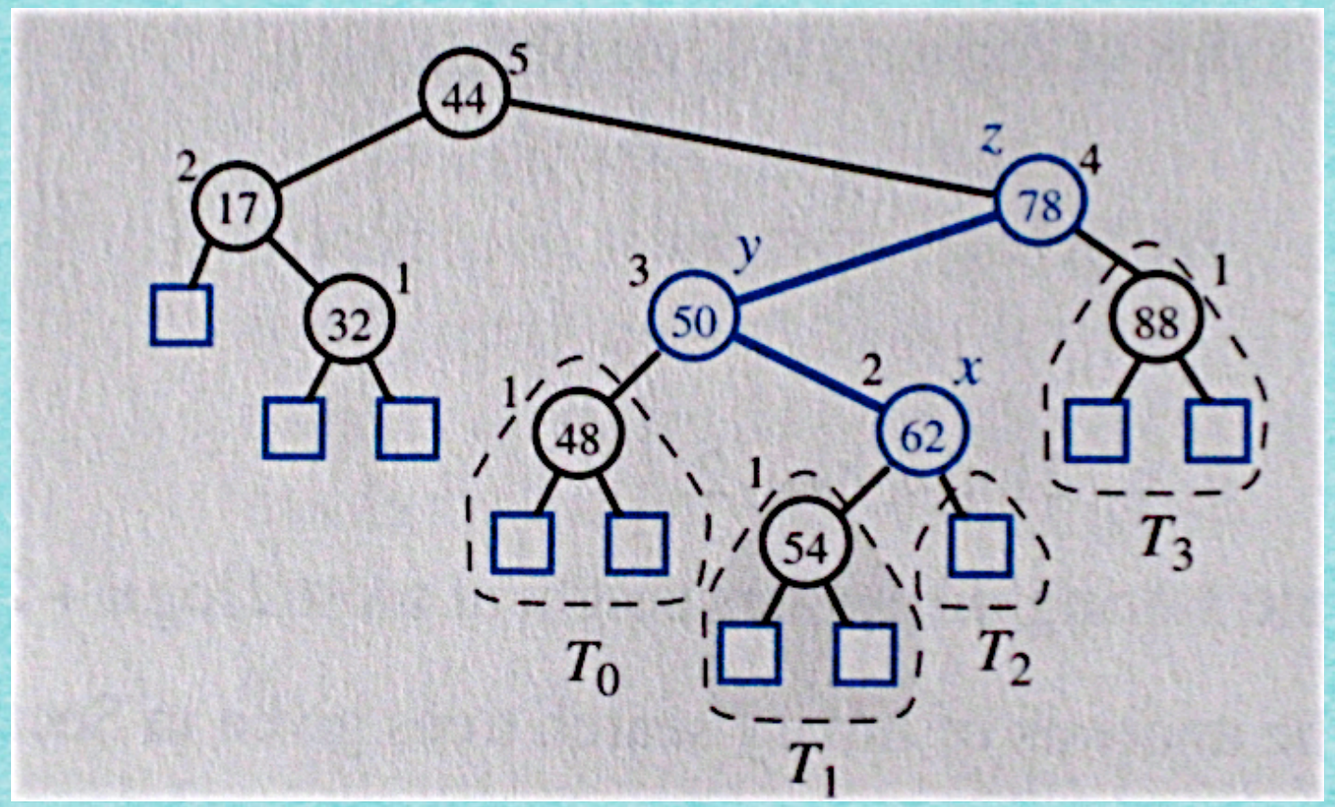
Einfügen

Idee:

- ***Höhenbalanciertheit ändert sich beim Einfügen einzelner Elemente nur wenig - und lokal!***



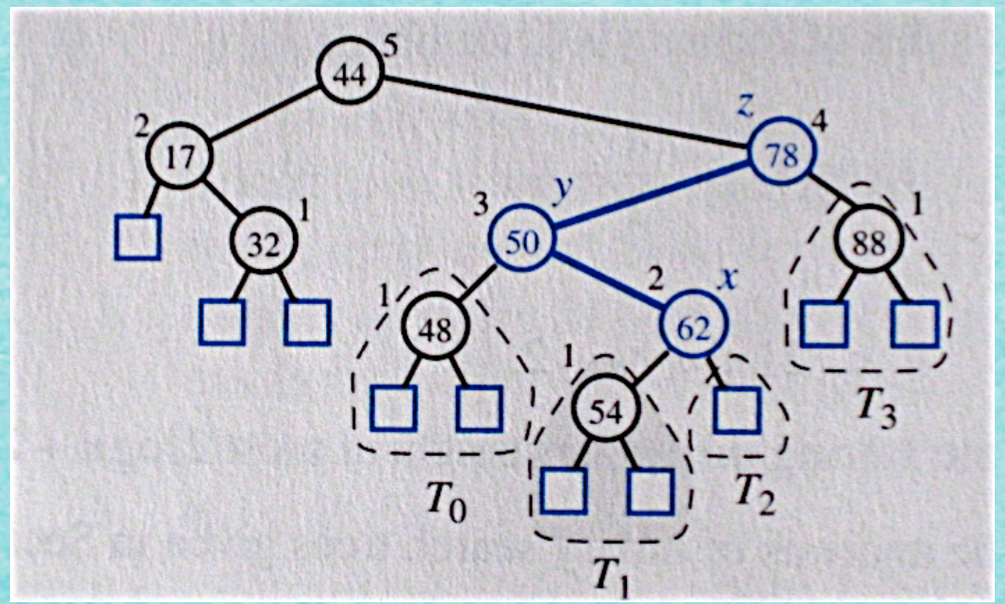
Einfügen

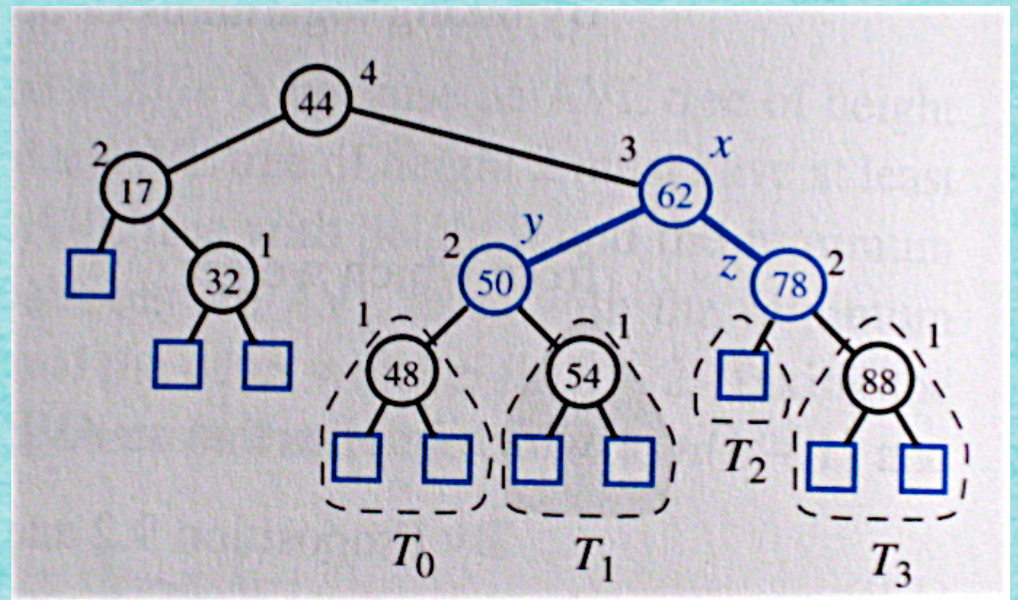


Was tun?

- ***Teilbaum der 78 ist nicht höhenbalanciert.***
- ***Die Höhe sollte höchstens 3 sein, damit auch der ganze Baum unter der 44 höhenbalanciert ist.***
- ***Betrachte Knoten 78, Kind 50, Enkel 62!***

Einfügen

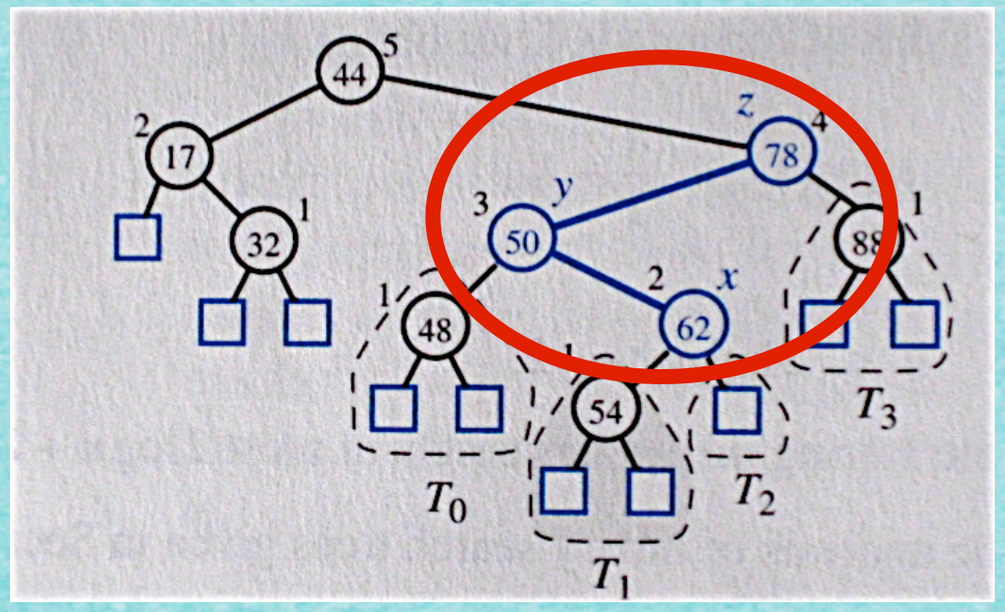


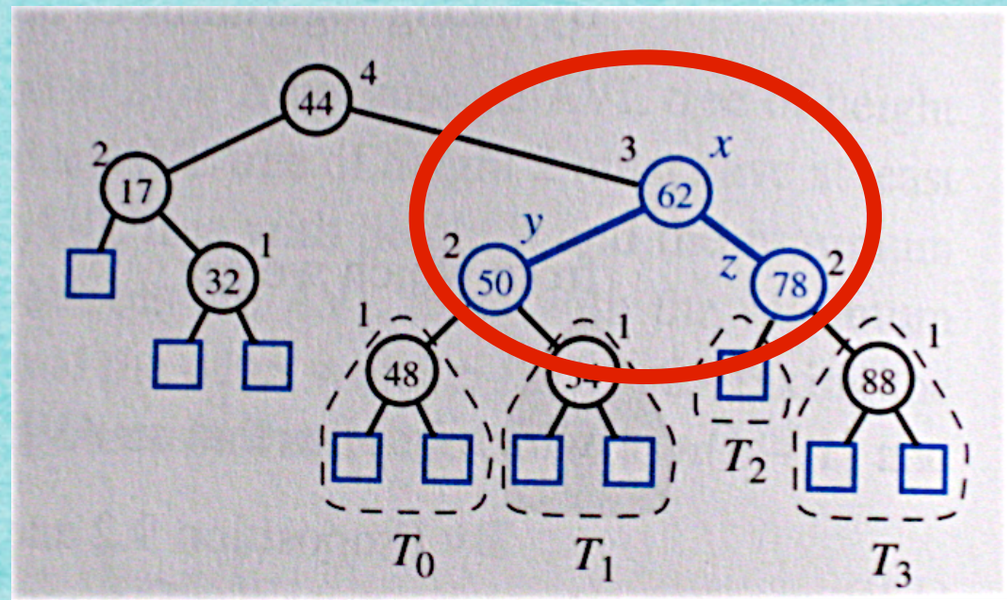


Neuer Baum!

- *Höhenbalanciert*
- *Nur lokale Umsetzung der Knoten 78, 50, 62*
- *Vorher drei Knoten untereinander, jetzt der mittlere über zwei anderen.*
- *„Rotation“*

Einfügen





Neuer Baum!

- *Höhenbalanciert*
- *Nur lokale Umsetzung der Knoten 78, 50, 62*
- *Vorher drei Knoten untereinander, jetzt der mittlere über zwei anderen.*
- *„Rotation“*

Algorithmus 4.9

INPUT:	Knoten x eines binären Suchbaumes T, Vaterknoten y, Großvaterknoten z
OUTPUT:	Binärer Suchbaum T nach Umstrukturierung mit x, y, z

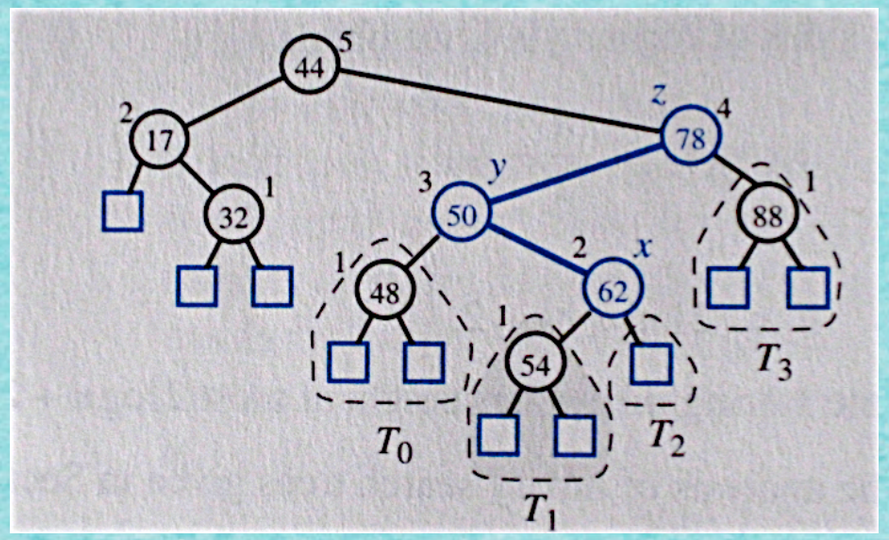
RESTRUCTURE(x)

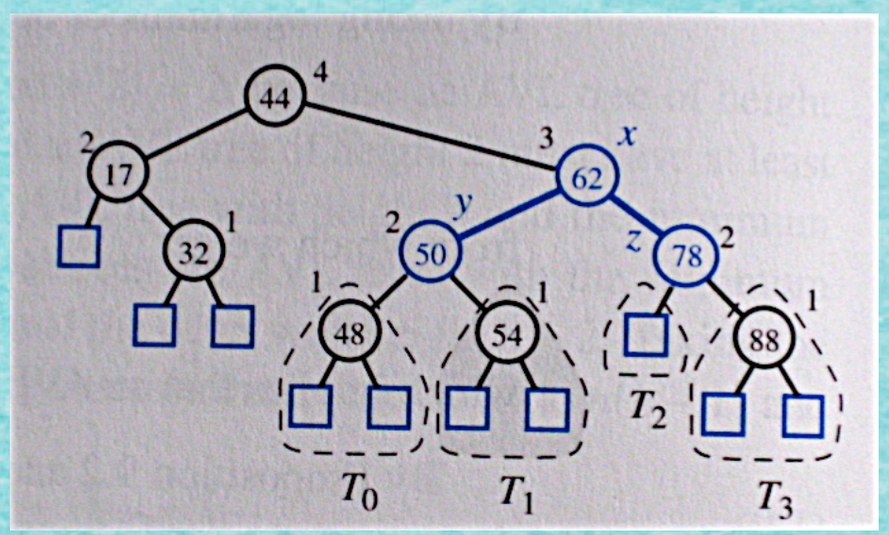
1. Sei (a, b, c) die Größensortierung der Knoten x, y, z;
   seien (T_0 , T_1 , T_2, T_3) die Größensortierung der vier Teilbäume unter x, y, z, die nicht Wurzeln x, y, z haben

2. Ersetze den Teilbaum mit Wurzel z durch einen neuen Teilbaum mit Wurzel b.

3. Setze a als linkes Kind von b, mit T_0 und T_1 als linken und rechten Teilbaum unter a;
   setze c als rechtes Kind von b, mit T_2 und T_3 als linken und rechten Teilbaum unter c.

4. RETURN



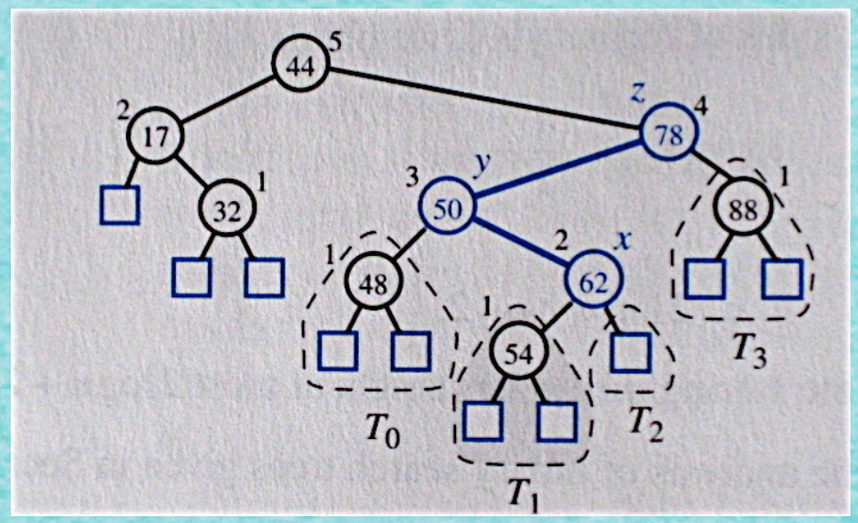


Satz 4.10

Mithilfe von RESTRUCTURE kann man einen AVL-Baum auch nach einer Einfüge-Operation höhenbalanciert halten.
Die Zeit dafür ist O(1).

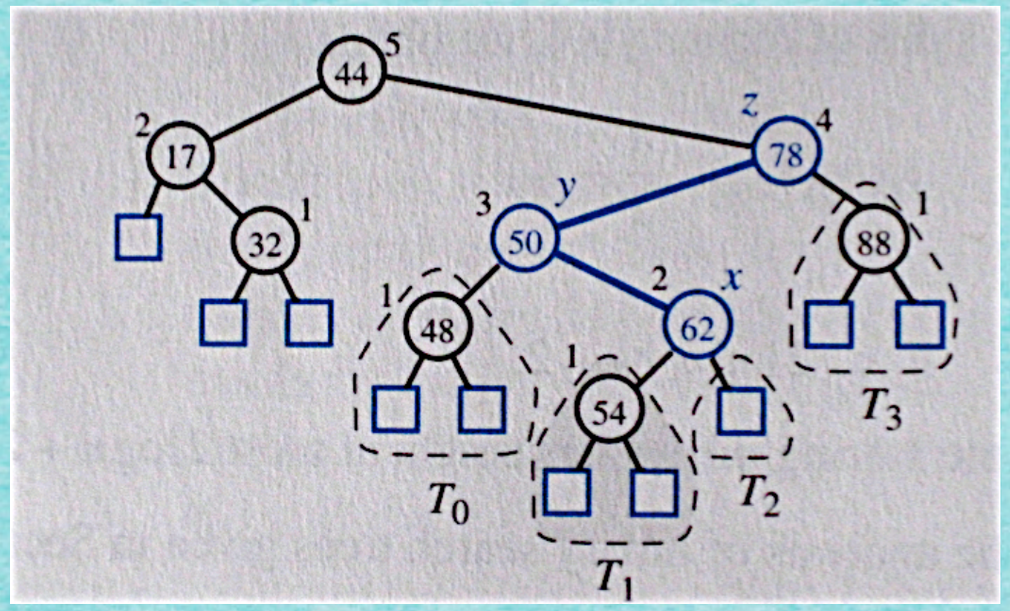
Beweis:

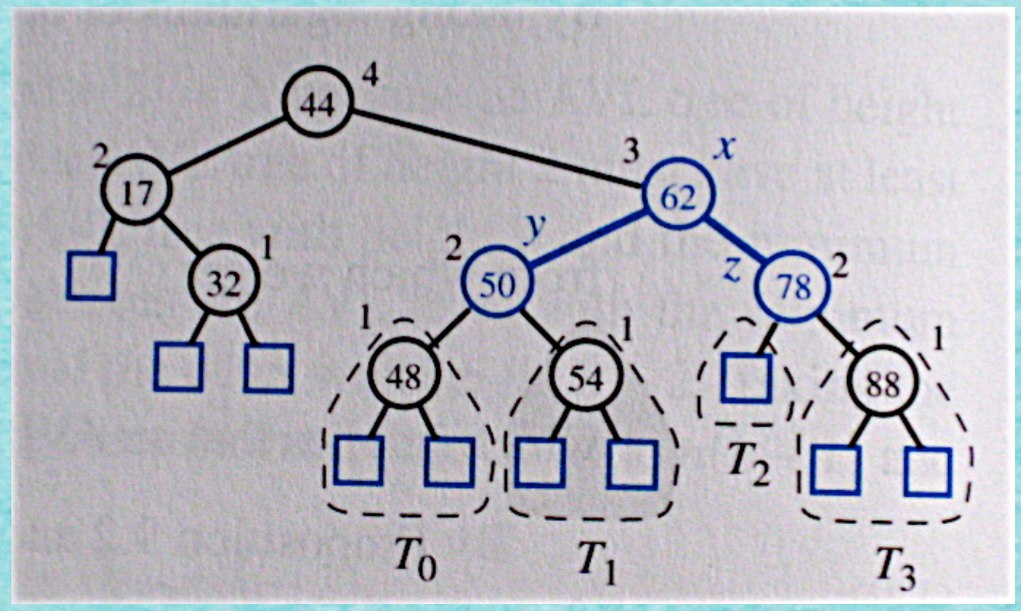
Angenommen, durch Hinzufügen eines Knotens v ist der Baum unbalanciert geworden.
Sei z der nach dem Einfügen niedrigste unbalancierte Vorfahre von v.
Sei y das Kind von z, das Vorfahre von v ist; y muss zwei höher sein als das andere Kind von z.
Sei x das Kind von y, das im selben Teilbaum wie v liegt.



Beweis von Satz 4.10 (Forts.):

Jetzt ersetzen wir die Teilstruktur z, y, x (3 Knoten untereinander) durch eine Teilstruktur mit 2 Knoten unter einem.
Z.z.: Danach ist der Baum ein AVL-Baum!



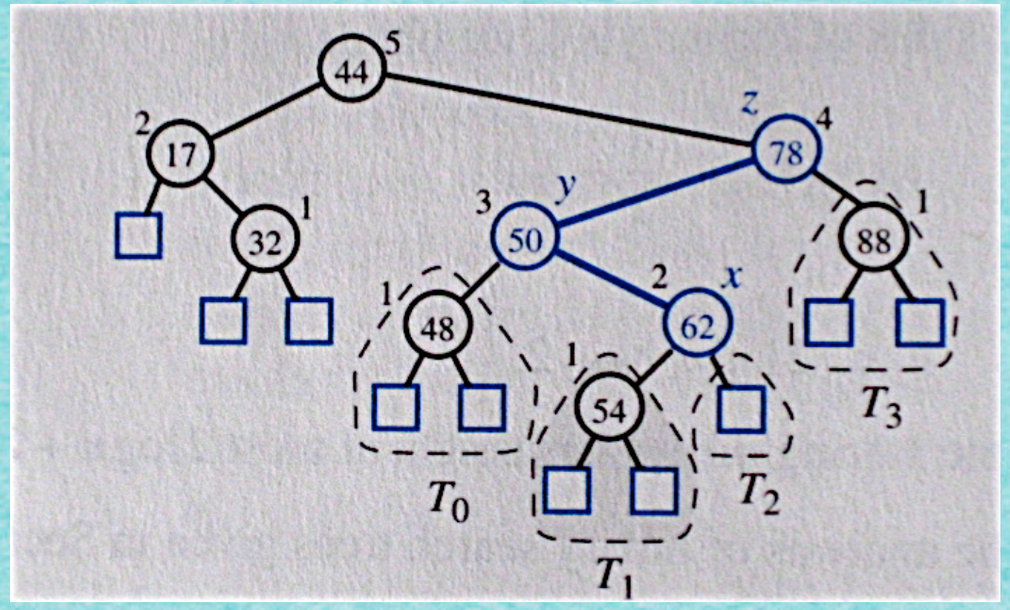


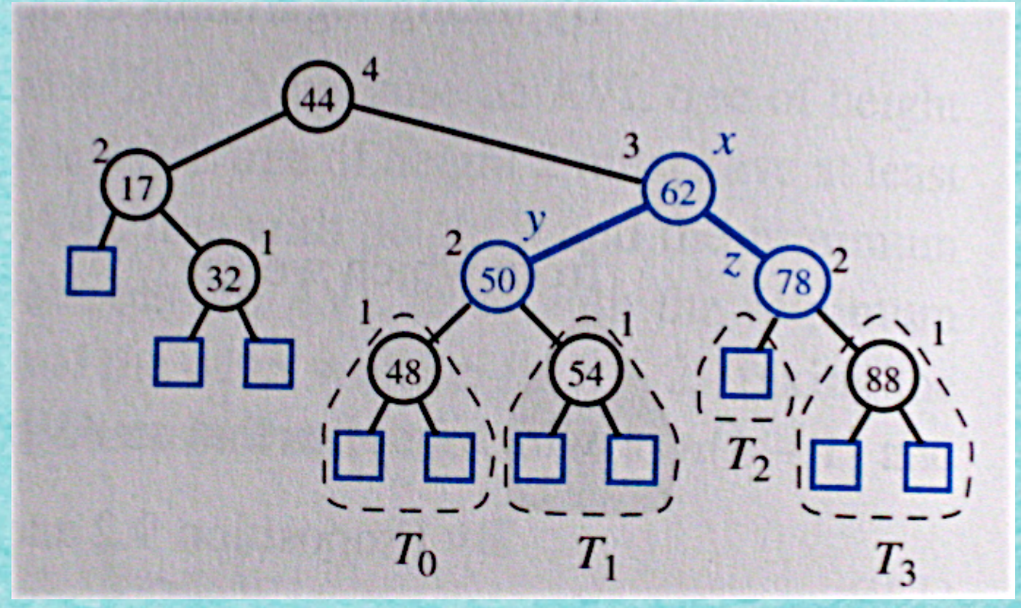
Betrachte jetzt die möglichen Anordnungen von x, y, z!

Beweis von Satz 4.10 (Forts.):

Welche Anordnungen gibt es?

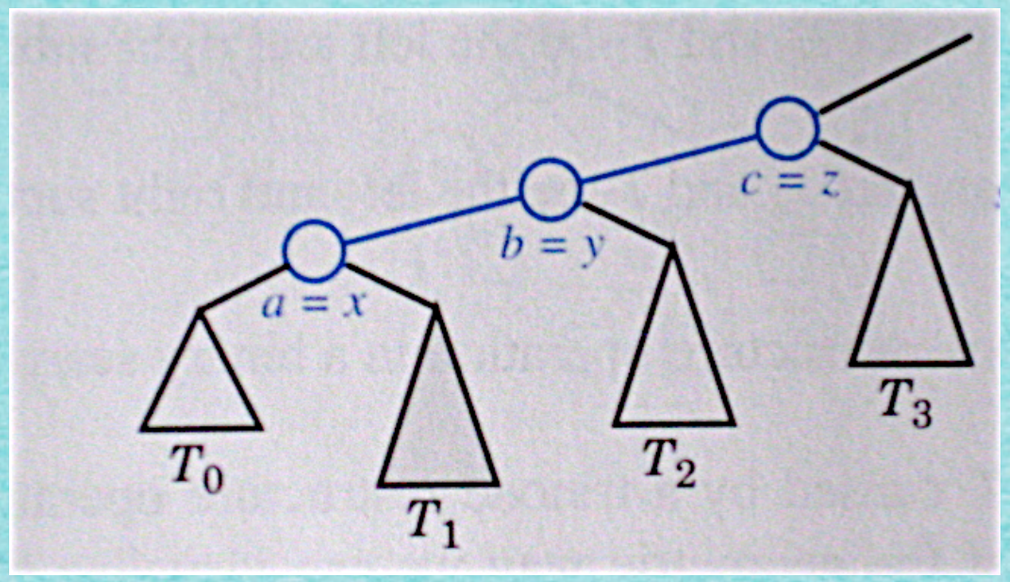
(1) $x \le y \le z$
(2) $x \le z \le y$
(3) $y \le x \le z$
(4) $y \le z \le x$
(5) $z \le x \le y$
(6) $z \le y \le x$

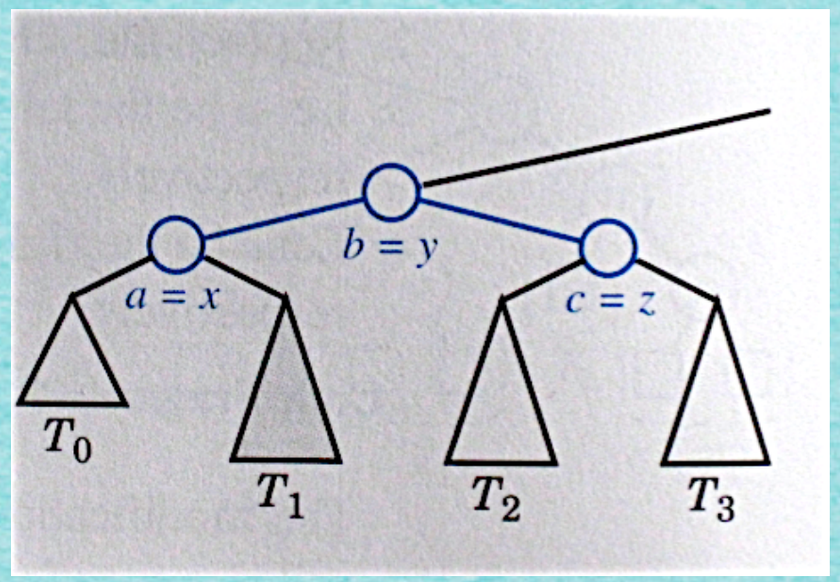




Beweis von Satz 4.10 (Forts.):

(1) $x \le y \le z$

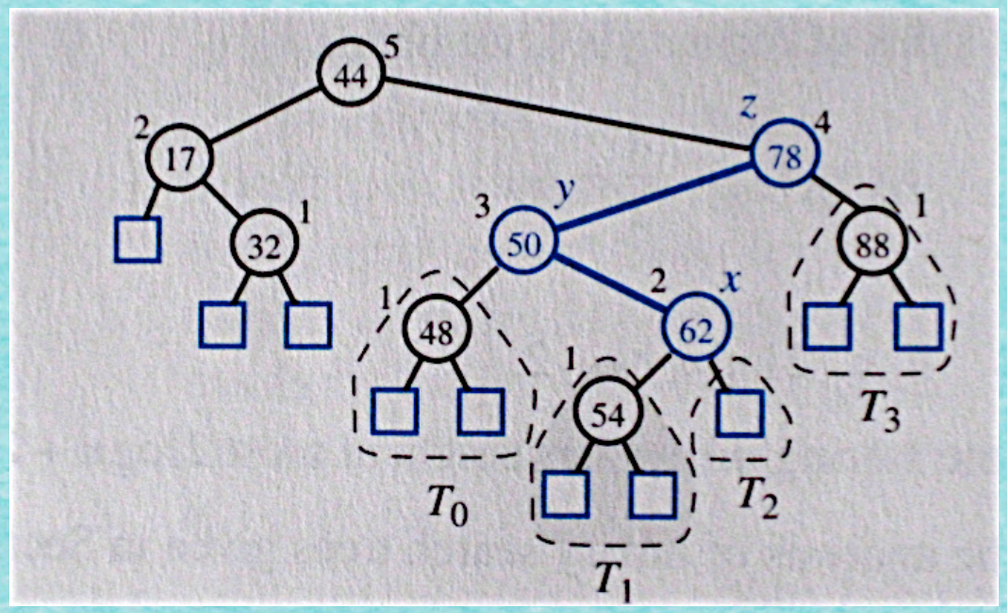




Der Baum ist wieder höhenbalanciert!

Beweis von Satz 4.10 (Forts.):

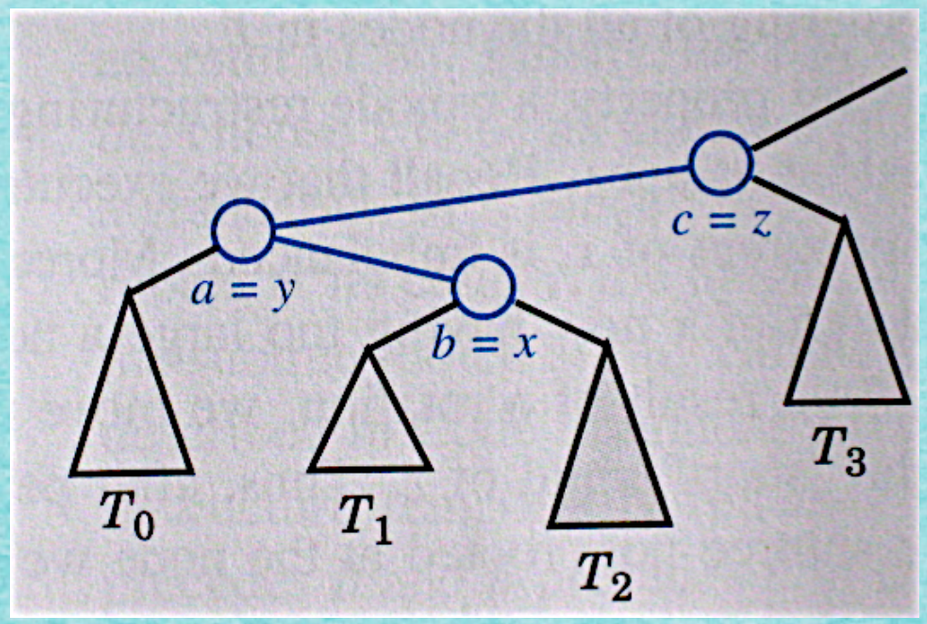
(2) $x \le z \le y$

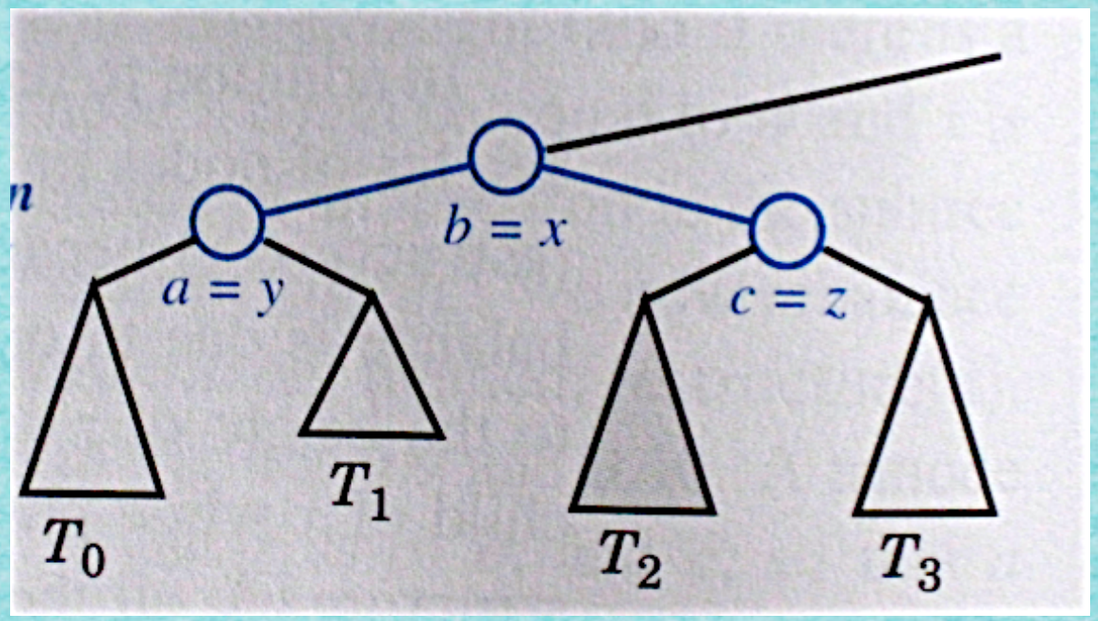


Der Fall kann nicht auftreten!

Beweis von Satz 4.10 (Forts.):

(3) $y \le x \le z$

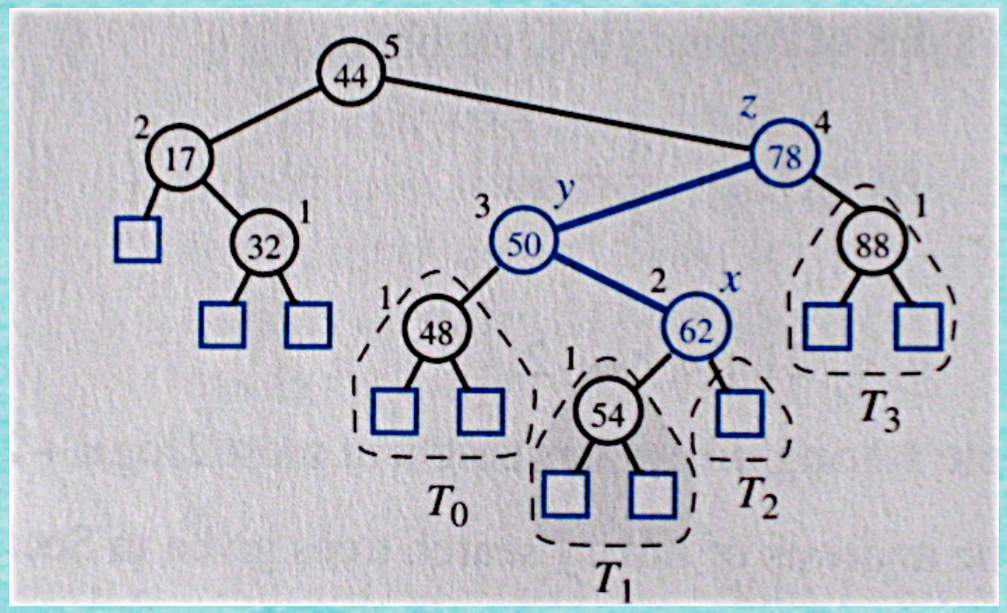




Der Baum ist wieder höhenbalanciert!

Beweis von Satz 4.10 (Forts.):

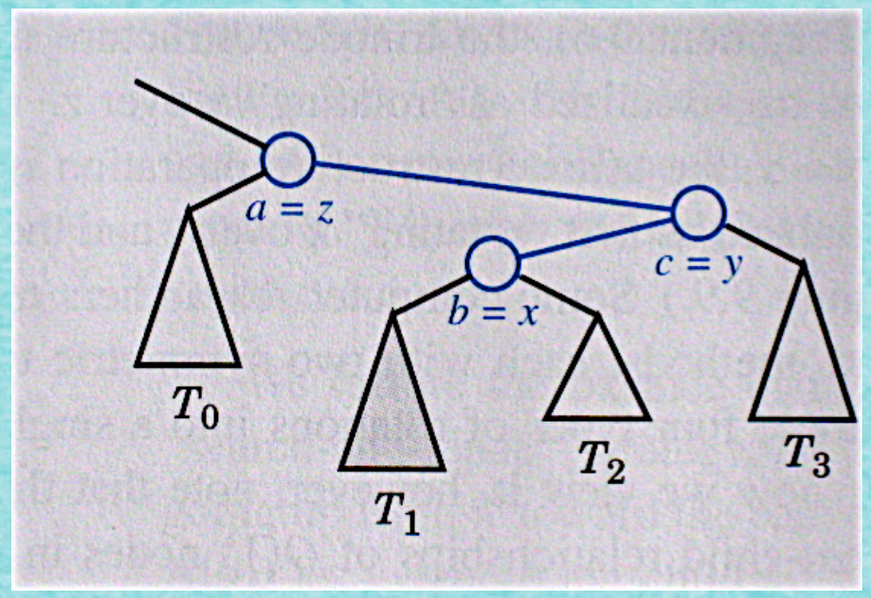
(4) $y \le z \le x$

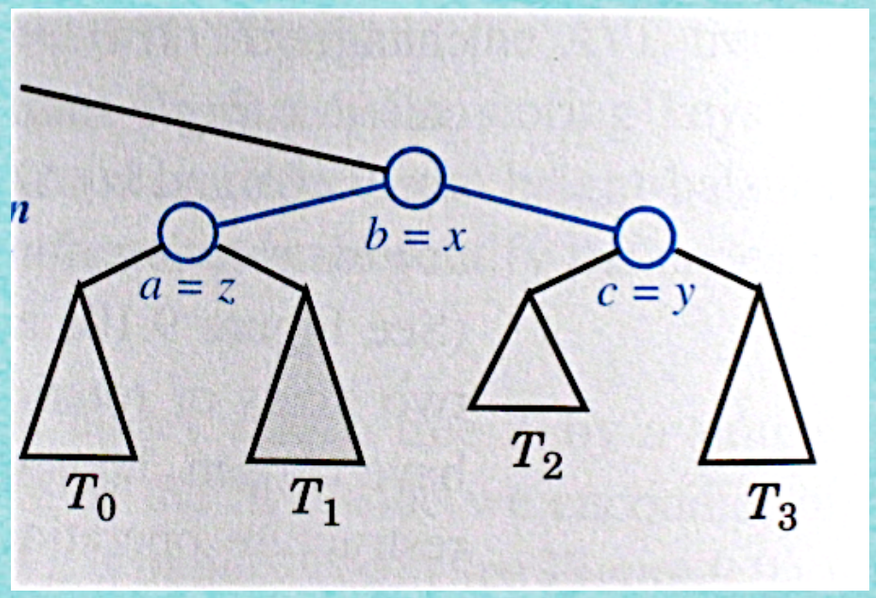


Der Fall kann nicht auftreten!

Beweis von Satz 4.10 (Forts.):

(5) z≤x≤y

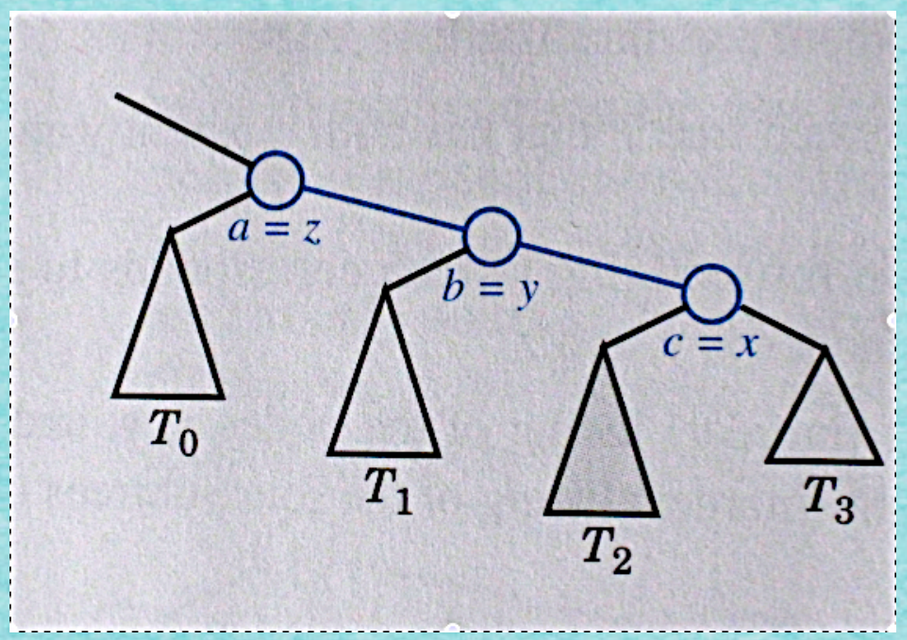


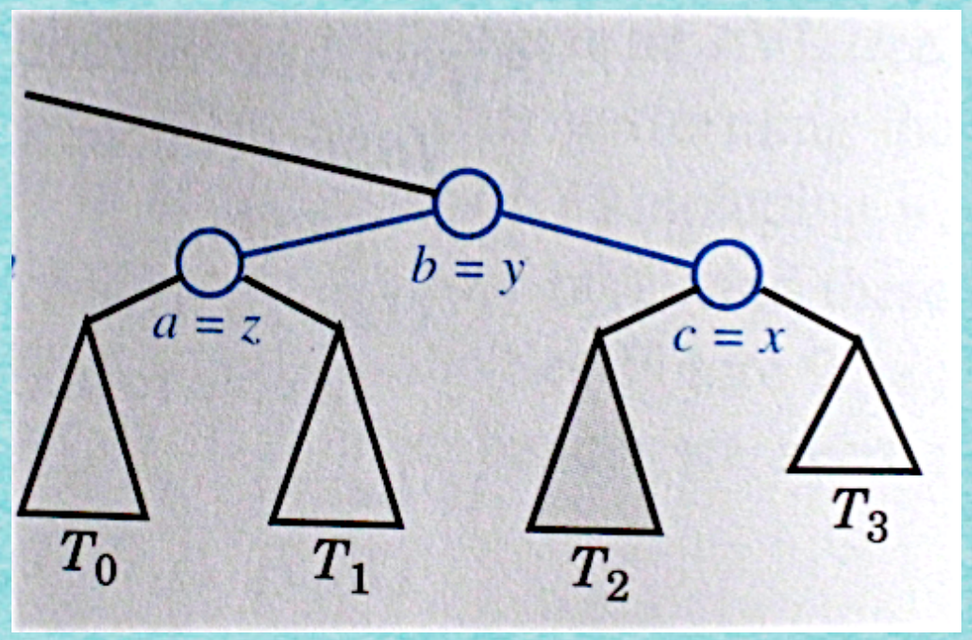


Der Baum ist wieder höhenbalanciert!

Beweis von Satz 4.10 (Forts.):

(6) z≤y≤x





Der Baum ist wieder höhenbalanciert!

Alle Schritte erfordern nur konstant viele Rechenoperationen.

Mehr demnächst!

s.fekete@tu-bs.de