FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Algorithmen und Datenstrukturen

Kapitel 5: Bäume Teil II

Prof. Ingrid Scholl

FH Aachen - FB 5

scholl@fh-aachen.de

27.04.2020

Übersicht dieser Vorlesung

- Ausgeglichene Bäume
  - 2-3-4 Suchbaum
  - Rot-Schwarz-Bäume

Bäume: Einleitung und Motivation

Bisher:

Kapitel 3

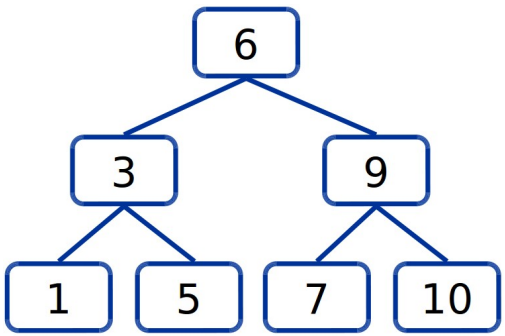
- Binärbäume
- Binäre Suchbäume
- Traversierungsarten in Bäumen

Kapitel 6

- Anwendung: Sortieren mit Bäumen (HeapSort)

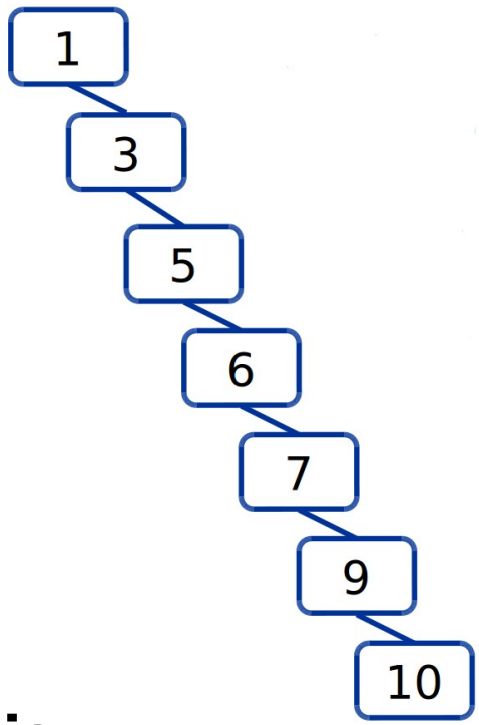
Gute und schlechte Suchbäume

Einfügereihenfolge:
6,3,9,1,5,7,10



Voll ausgeglichener Baum!
Höhe $= 2$

Einfügereihenfolge:
1,3,5,6,7,9,10



Ergebnis:
verkettete Liste, entarteter Baum! Höhe $= 6$

Bäume

Höhe eines Baumes mit n-Elementen:

- Worst Case: Höhe $h = n$, entarteter Baum
- Best Case: Höhe $h = \log_2 n$, ausgeglichener Baum

Definition (Balancierte Bäume)

Suchbäume mit einer logarithmischen Höhe nennt man ausgeglichene oder balancierte Bäume.

Ziel:

- möglichst geringe Höhe bei n Knoten
- Bäume möglichst ausgeglichen lassen

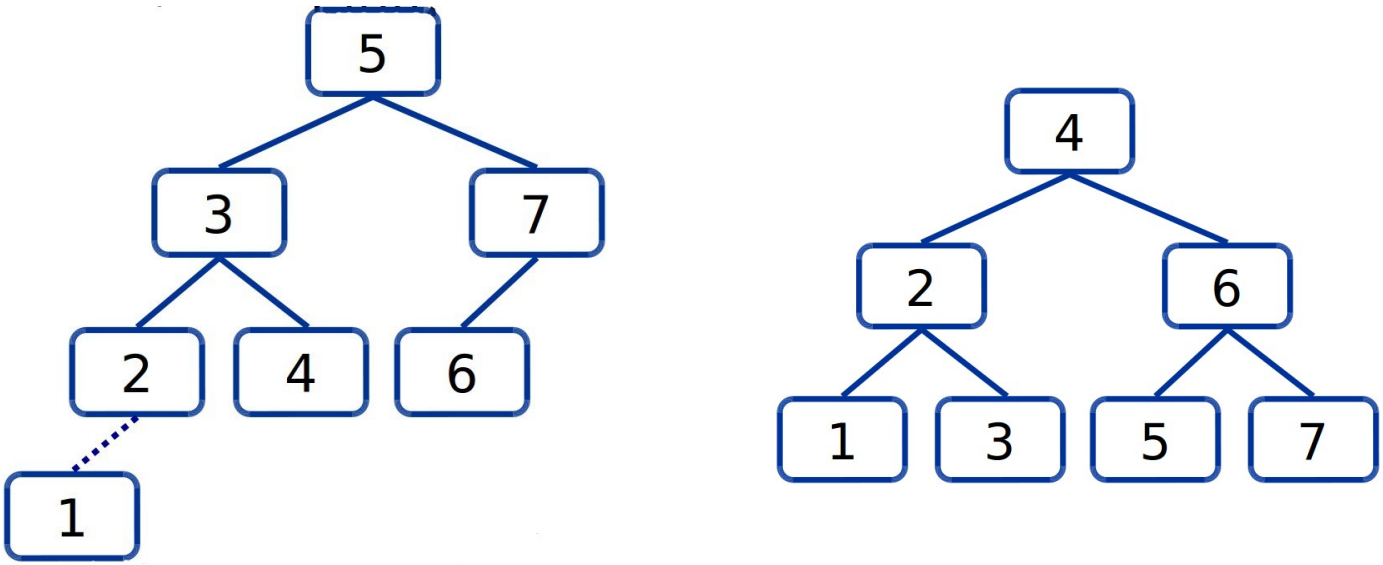
Fragen:

- Wann kommt es bei Bäumen zu strukturellen Änderungen?
- Bei welchen Operationen verändert sich die Höhe des Baumes?

Wie erreicht man ausgeglichene Bäume?

Idee: Baum nach jeder Einfüge- und Löschoperation ausgleichen.

Beispiel: Einfügen von 1 in einem binären Suchbaum



Nachteil: Worst Case - Bewegung von jedem Knoten!

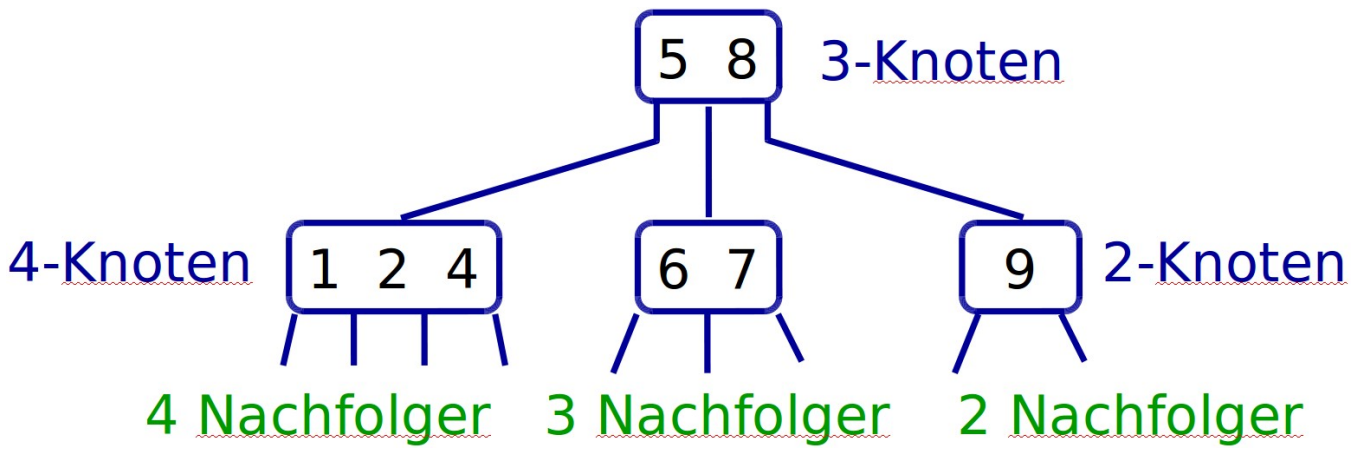
Wie erreicht man ausgeglichene Bäume?

1. 2-3-4-Suchbaum oder Rot-Schwarz-Baum:
   mehr Flexibilität beim Einfügen durch Knoten mit mehr als einem Schlüssel (d.h. ≥ 2 Nachfolger)
2. B-Baum:
   besitzen eine ausgeglichene Höhe, lassen aber einen unausgeglichenen Verzweigungsgrad zu (spez. in DB-Anwendungen als Indexstruktur)
3. AVL-Baum:
   abgeschwächtes Kriterium für eine ausgeglichene Höhe (Balance-Kriterium)

2-3-4 Suchbaum

Idee: Knoten mit mehr Flexibilität

- Knoten mit 1 Schlüssel: 2 Nachfolger (2-Knoten)
- Knoten mit 2 Schlüsseln: 3 Nachfolger (3-Knoten)
- Knoten mit 3 Schlüsseln: 4 Nachfolger (4-Knoten)



2-3-4 Suchbaum

Idee:

- Größere Flexibilität bei der Schlüsselanzahl pro Knoten
- ≥ 1 Schlüssel und ≤ 3 Schlüssel pro Knoten sind erlaubt
- Strukturänderungen des Baumes entstehen durch das Einfügen und das Löschen eines Schlüssels und sind nur auf einen Teilbereich des Baumes beschränkt, was den Aufwand für das Ausgleichen reduziert

2-3-4 Suchbaum

Definition (2-3-4-Suchbaum)

Ein 2-3-4-Suchbaum ist ein Baum, der entweder leer ist oder

- ein 2-Kind-Knoten mit einem Schlüssel und zwei Referenzen: einer linken Referenz zu einem 2-3-4-Suchbaum mit kleineren Schlüsseln und einer rechten Referenz zu einem 2-3-4-Suchbaum mit größeren Schlüsseln.
- ein 3-Kind-Knoten mit zwei sortierten Schlüsseln und 3 Referenzen.
- ein 4-Kind-Knoten mit drei sortierten Schlüsseln und 4 Referenzen.
- eine Referenz zwischen 2 Schlüsselwerten a und b verweist auf einen 2-3-4-Suchbaum mit Schlüsselwerten key für die gilt:

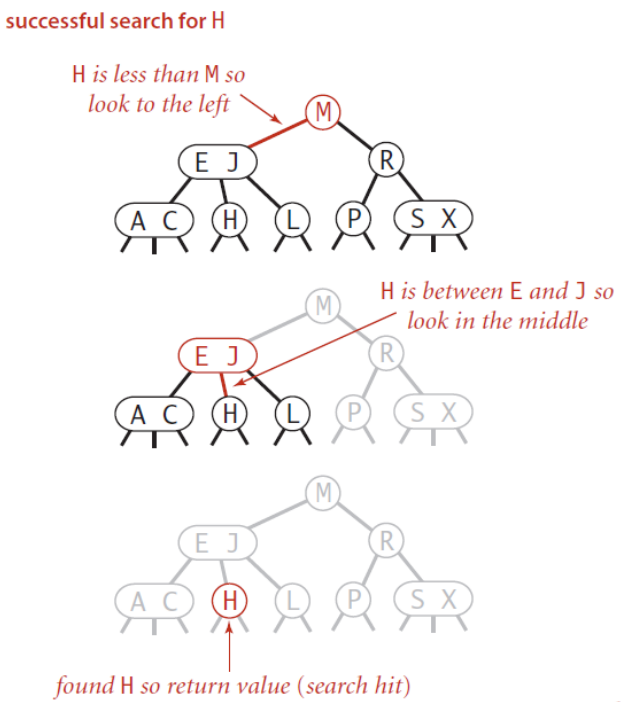
$$a < key < b$$

2-3-4 Suchbaum

Wie funktionieren die Grundoperationen?
Zunächst erst mal:

- Suchen
- Einfügen

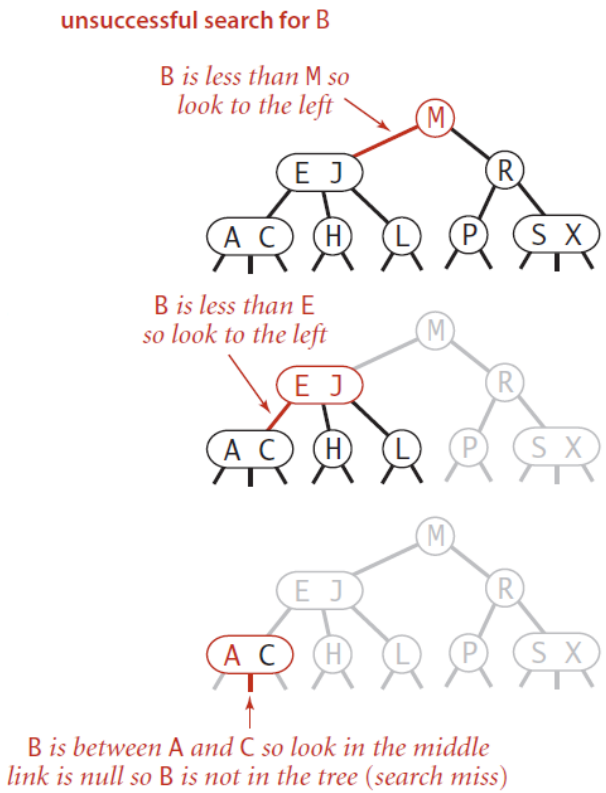
Suche im 234-Suchbaum



Suchalgorithmus 234-Suchbaum:

- Analog zur Suche im BST.
- Vergleiche Suchschlüssel mit Schlüsseln in der Wurzel: Stimmt einer von diesen überein, dann war die Suche erfolgreich. Andernfalls wird diejenige Referenz weiter untersucht, dessen Schlüsselwerte-Intervall den Suchschlüssel enthalten könnte...

Suche im 234-Suchbaum



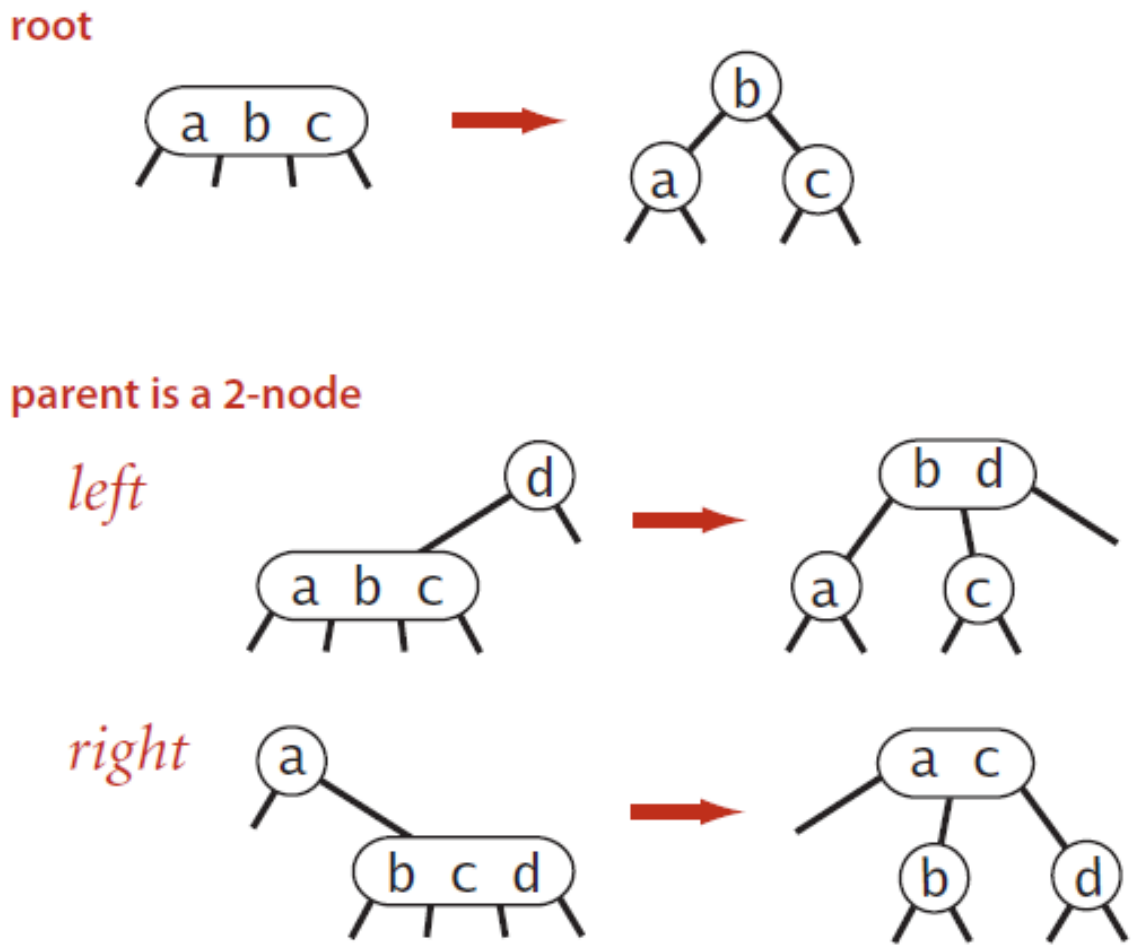
Suchalgorithmus 234-Suchbaum:

- Analog zur Suche im BST.
- Vergleiche Suchschlüssel mit Schlüsseln in der Wurzel: Stimmt einer von diesen überein, dann war die Suche erfolgreich. Andernfalls wird diejenige Referenz weiter untersucht, dessen Schlüsselwerte-Intervall den Suchschlüssel enthalten könnte. Ist die Referenz *null*, dann war die Suche erfolglos.

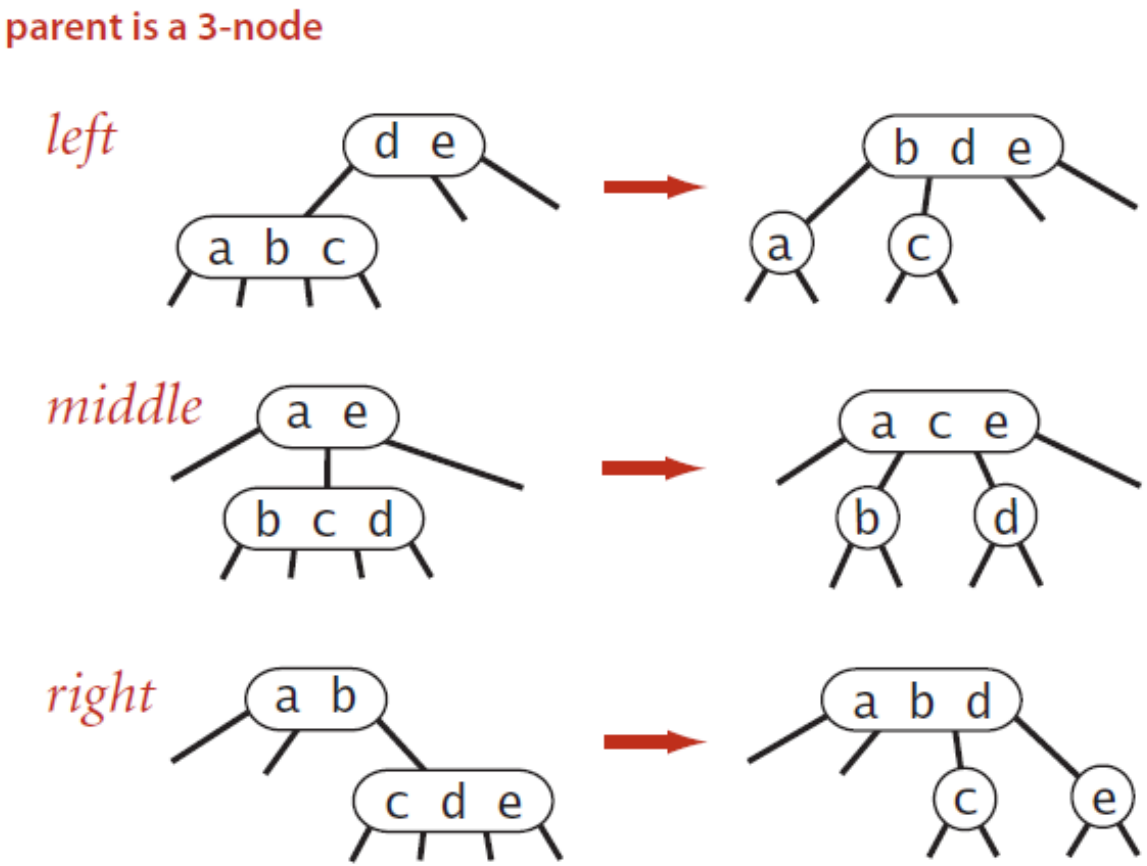
Einfügen im 2-3-4 Suchbaum

1. Einfügeposition ermitteln (Einfügen im Blattknoten)
2. Fallunterscheidung:
   - 2.1 Einfügen im 2- oder 3-Blattknoten:
     Knoten wird um das neue Element im Blattknoten erweitert, der Schlüssel wird sortiert eingefügt.
   - 2.2 Einfügen im 4-Knoten:
     4-Knoten wird in zwei 2-Knoten aufgeteilt (Split-Methode)

Splitten eines 4-er Knoten



Splitten eines 4-er Knoten



Einfügen im 2-3-4 Suchbaum

Aufsplitten eines 4-Knoten in zwei 2-Knoten:

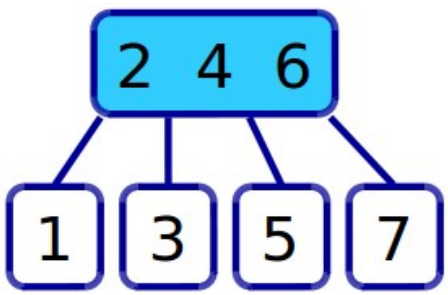
- **Bottom-Up:**

  Das mittlere Schlüsselelement vom Blattknoten wird zum Elternknoten verschoben. Ist dieser wieder ein 4-Knoten, muss dieser Knoten auch aufgeteilt werden, ... (rekursive Methode)

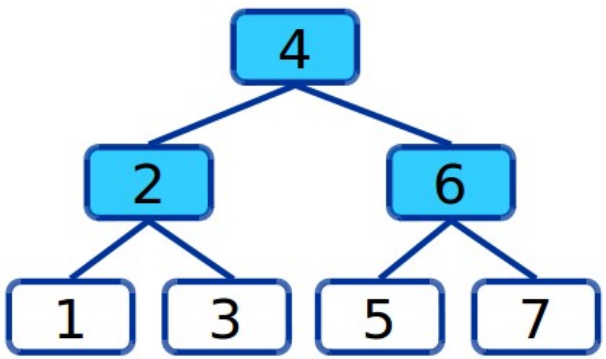
- **Top-Down** (günstiger):

  Auf dem Weg von der Wurzel bis zur Einfügeposition (Blattknoten) werden alle besuchten 4-Knoten vorsorglich in 2-Knoten umgewandelt (iterative Methode)

4-Knoten:



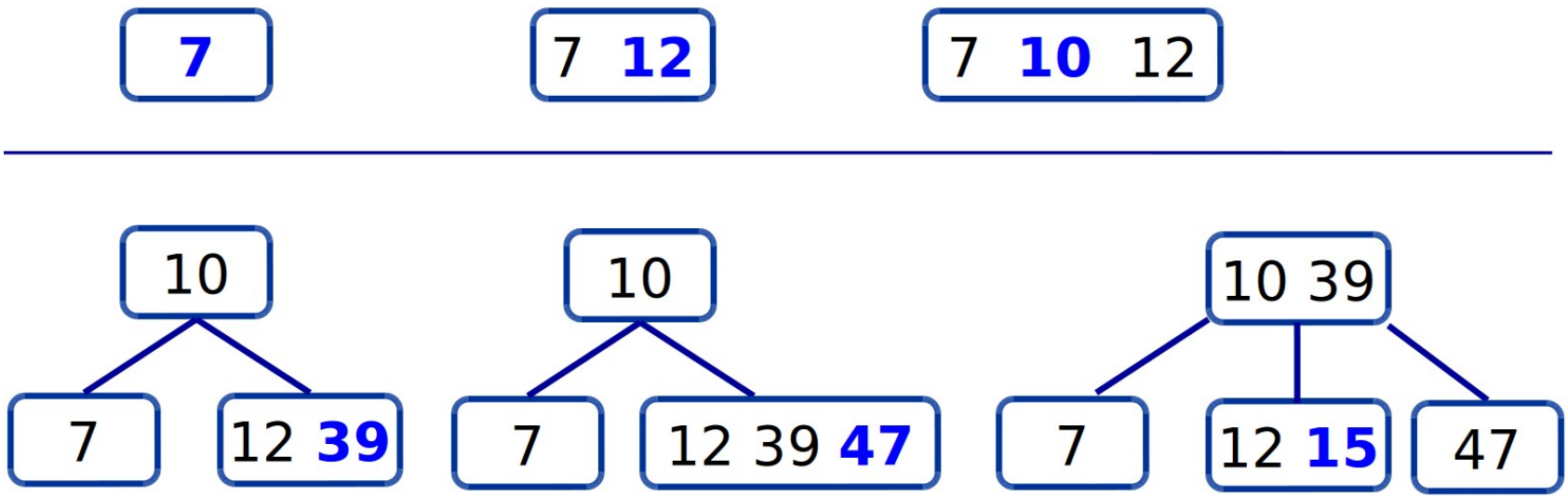
2-Knoten:



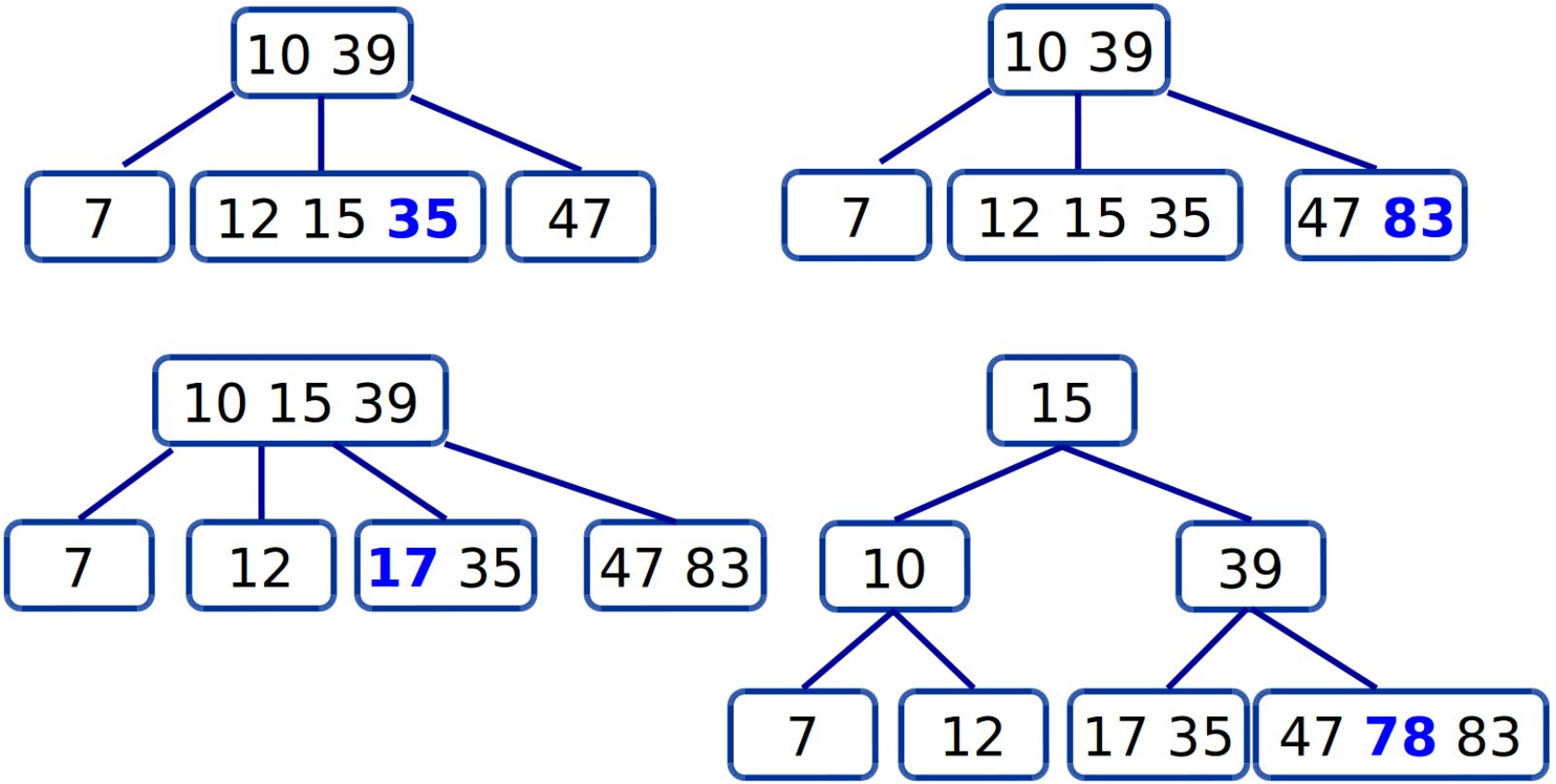
Beispiel: Top-Down

Gegeben: 7,12,10,39,47,15,35,83,17,78

Gesucht: Top-Down 2-3-4 Suchbaum



Beispiel: Top-Down



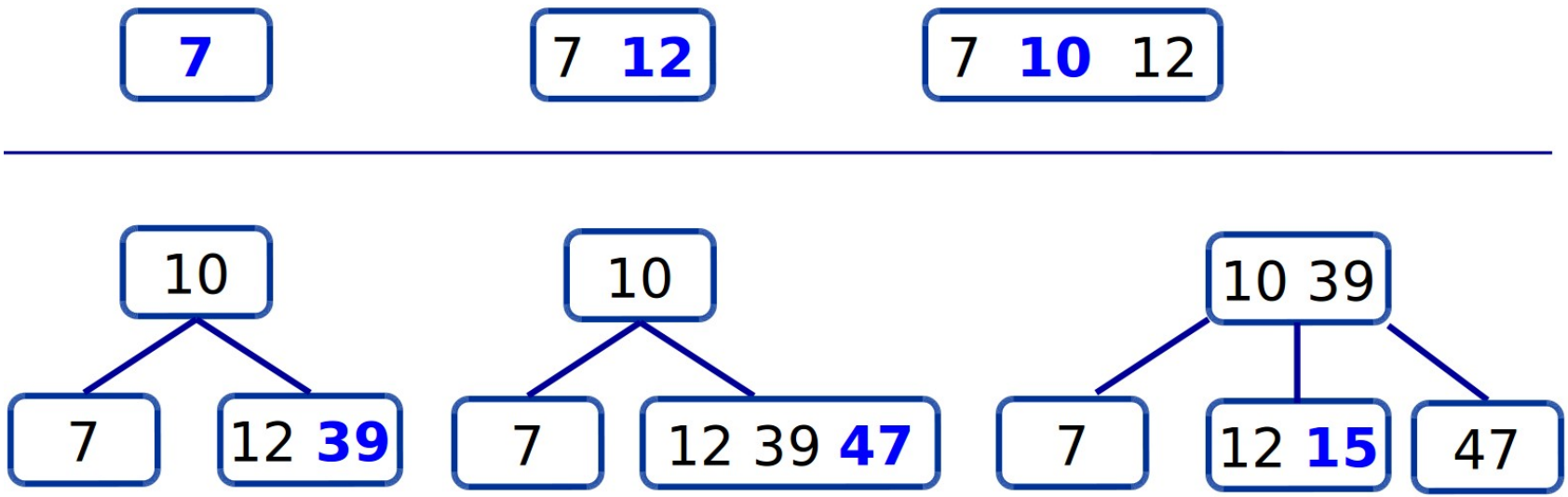
Beispiel: Bottom-Up

- Erst Einfügeposition suchen (Blattelement)
- Ist dieser Blattknoten ein 4-Knoten, dann wird dieser in 2 2-Knoten umgewandelt, indem der mittlere Schlüssel zum Elternknoten hinzugefügt wird.
- Falls der Elternknoten auch ein 4-Knoten ist, muss dieser auch umgewandelt werden usw.
- d.h. rekursive Umwandlung der 4-Knoten von „unten nach oben“ - eben Bottom-Up, es kommt ggfls. in der Wurzel zu einem Höhenzuwachs des Baumes um 1.

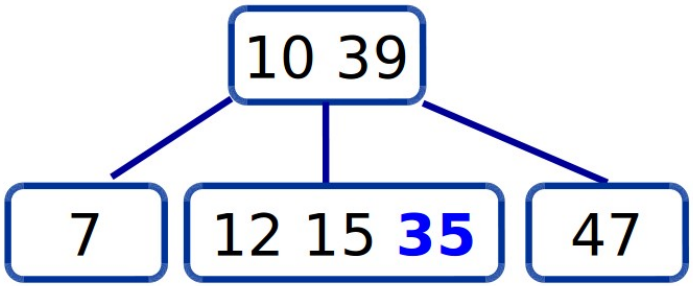
Beispiel: Bottom-Up

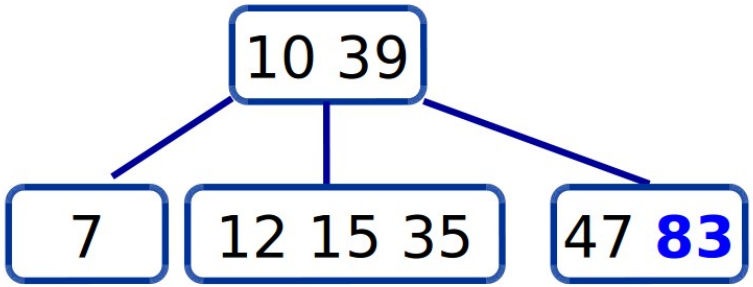
Gegeben: 7,12,10,39,47,15,35,83,17,78

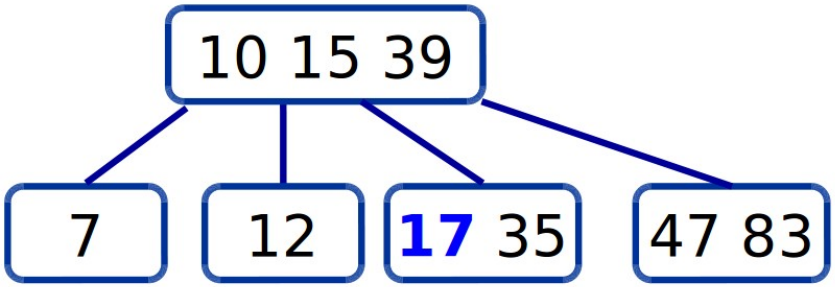
Gesucht: Bottom-Up 2-3-4 Suchbaum



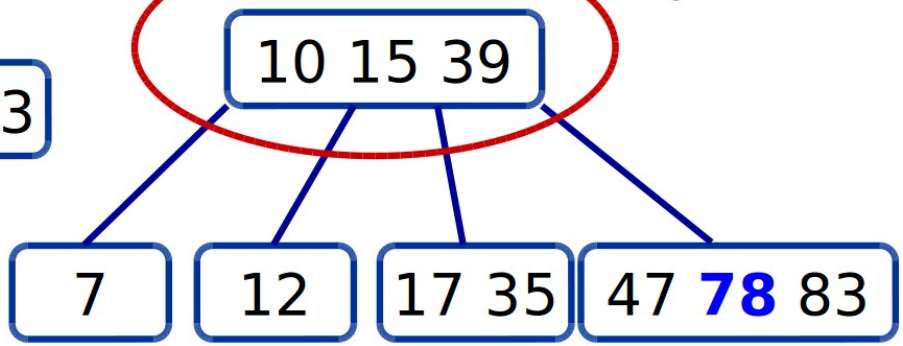
Beispiel: Bottom-Up





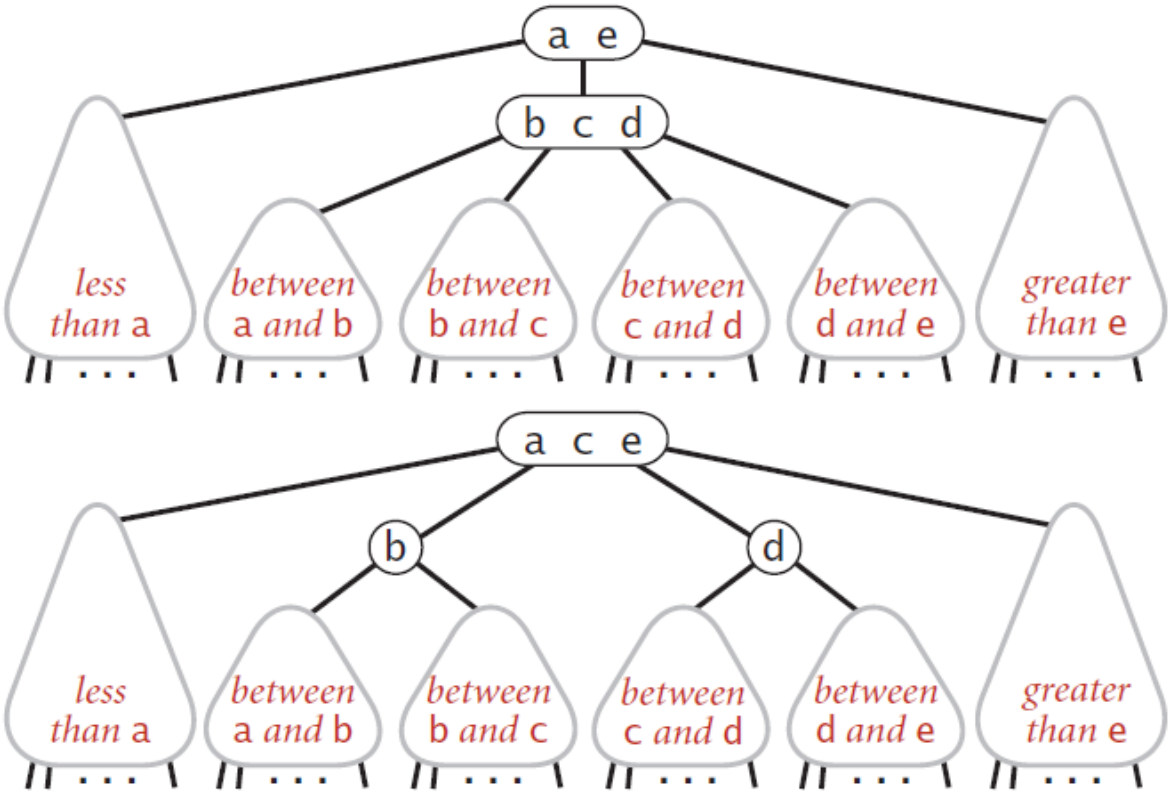


Unterschied zu Top Down



Lokale Transformation

Das Splitten (Teilen) eines 4-Kind-Knotens ist eine lokale Transformation, die die Ordnung und die perfekte Balance erhält.



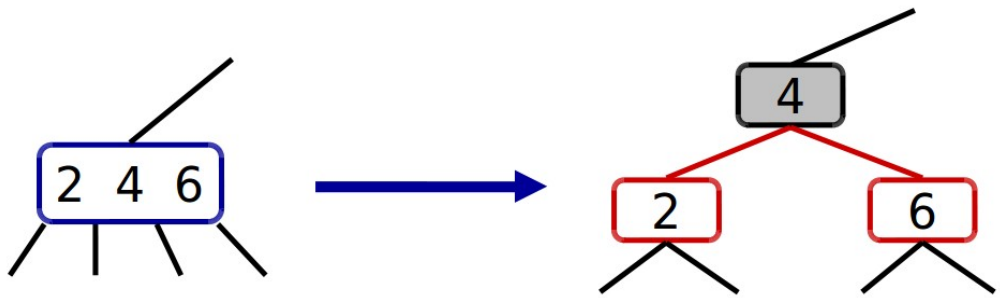
Rot-Schwarz-Bäume

Implementierung des 2-3-4 Suchbaumes als Rot-Schwarz-Baum (spezieller Binärbaum):

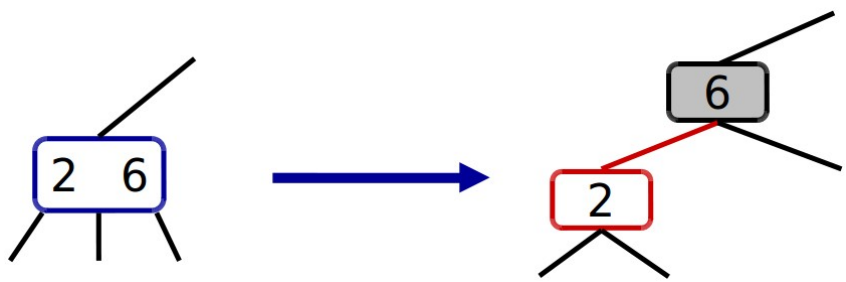
- 3- und 4-Knoten werden als kleine Binärbäume dargestellt.
- Knoten (Kanten) erhalten eine Farbe $color \in \{red, black\}$. Ein Knoten (Kante) ist rot, wenn dieser zu seinem Elternknoten gehört, andernfalls schwarz.
- Scharze Knoten sind diejenigen Knoten des ursprünglichen 2-3-4-Baumes.
- Rote Knoten sind diejenigen Knoten, die im 2-3-4-Baum zu ihrem Elternknoten gehören.
- Es wird für einen Knoten die Kante zum Elternknoten eingefärbt.

Rot-Schwarz-Bäume

4-Knoten im Rot-Schwarz-Baum:



3-Knoten im Rot-Schwarz-Baum:

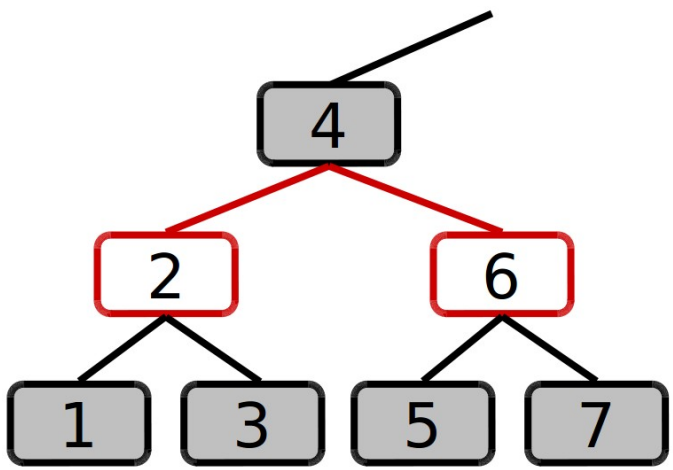


Knoten haben die Farbe der Kante zu ihrem Elternknoten.

Rot-Schwarz-Bäume

- Farbe einer Kante (rot oder schwarz) wird im Knoten gespeichert
- Knoten bekommt die Farbe der Kante zum Elternknoten
- **Ergebnis:**
  rote und schwarze Knoten!

Beispiel:



Rot-Schwarz-Bäume

Definition (Rot-Schwarz-Bäume)

Ein Rot-Schwarz-Baum ist ein binärer Suchbaum mit folgenden (Rot-Schwarz-) Eigenschaften:

1. Jeder Knoten ist entweder rot oder schwarz.
2. Jeder neu einzufügende Blattknoten ist rot.
3. Die Kinder von einem roten Knoten sind schwarz.
4. Es gibt **keine zwei aufeinanderfolgende rote Knoten**.
5. **Kriterium für Ausgeglichenheit:**
   Für jeden Knoten *k* gilt: Jeder Pfad von *k* zu einem Blatt enthält die gleiche Anzahl schwarzer Knoten.
6. Die Wurzel ist immer schwarz.

Rot-Schwarz-Bäume

Grundoperationen im Rot-Schwarz-Baum:

- Suchen analog zum binären Suchbaum.
- Traversierung analog zum binären Suchbaum, jedoch zusammenhängende Knoten beachten.
- Einfügen ist etwas komplizierter, da hierbei die Rot-Schwarz-Eigenschaften erhalten bleiben müssen:
  - Einfügen beim 2-3-4 Suchbaum: Splitten der 4-Knoten bottom-up oder top-down (Hier: nur top-down)
  - Ggf. Umstrukturierung durchführen durch **Rotationen** von Knoten.

Top-Down Einfügen

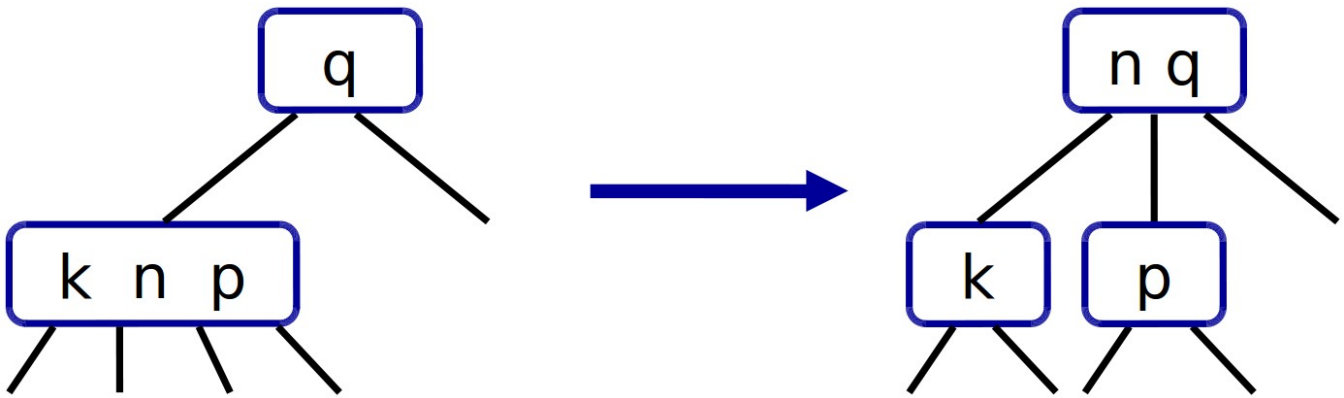
Ziel: Splitten des 4-Knoten.

Fallunterscheidung:

1. 4-Knoten hängt am 2-Knoten
2. 4-Knoten hängt am 3-Knoten
   a) Ist das linke äußere Kind des 3-Knoten
   b) Ist das mittlere Kind des 3-Knoten
   c) ist das rechte äußere Kind des 3-Knoten

Top-Down Einfügen - Fall 1

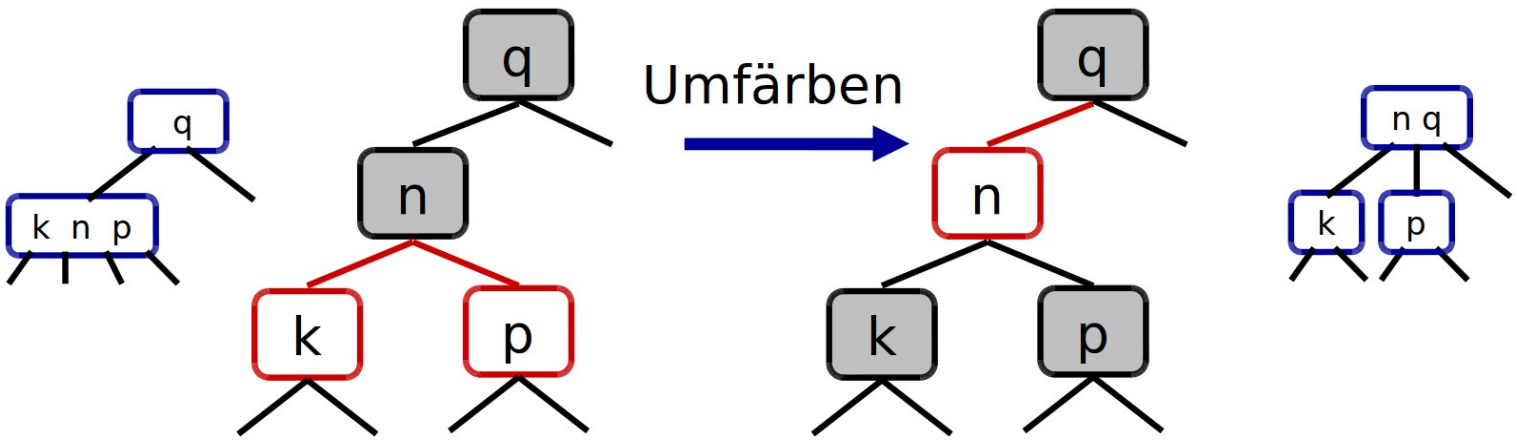
4-Knoten hängt an 2-Knoten



Umwandlung des Elternknoten *q* in einen 3-Knoten

Top-Down Einfügen - Fall 1

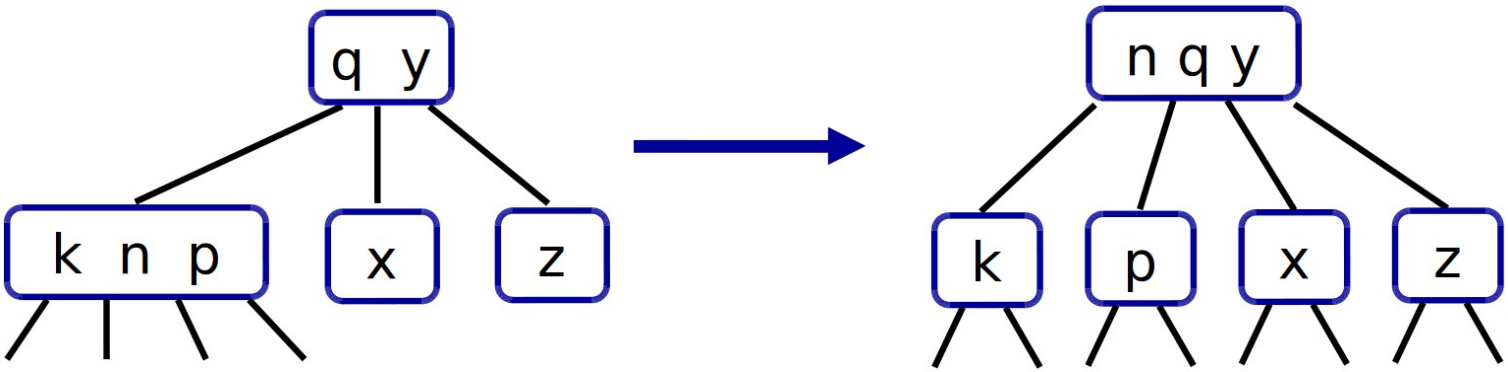
Umwandlung des Elternknoten (*q*) (2-Knoten) in einen 3-Knoten (*nq*) entspricht dem Umfärben der Knoten.



Top-Down Einfügen - Fall 2a (und c)

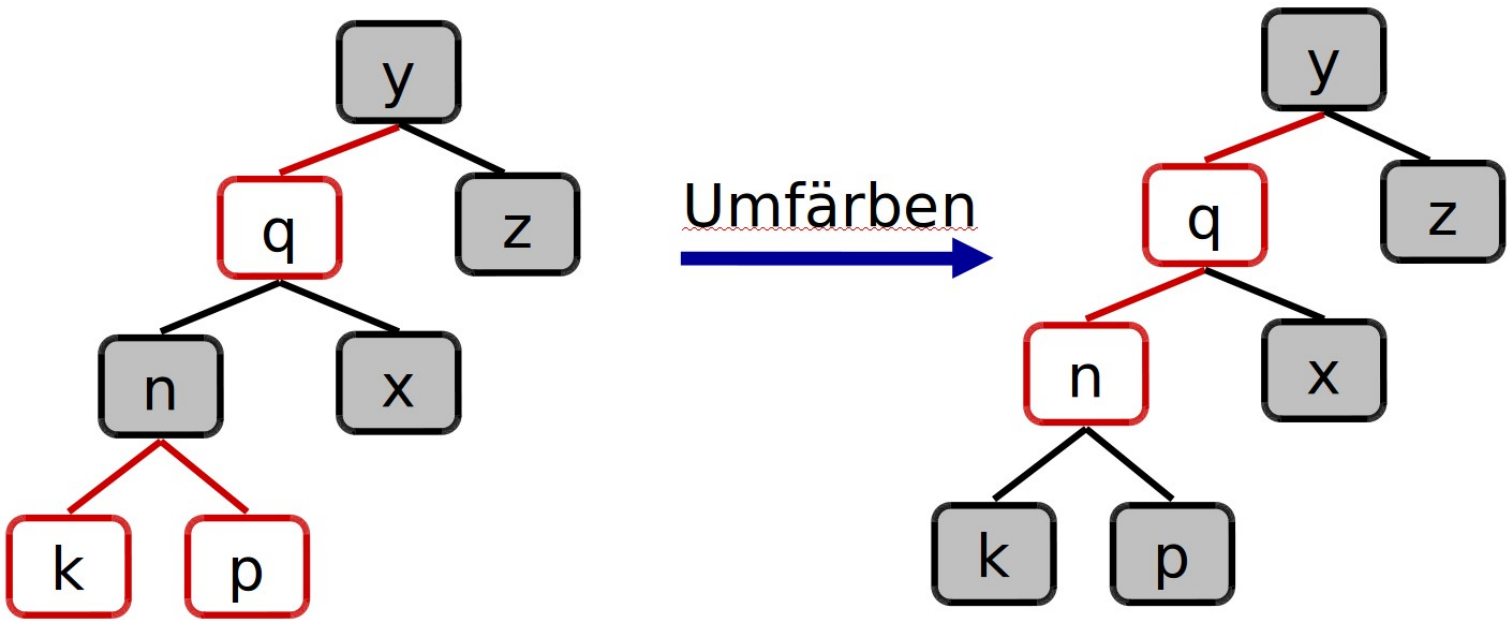
4-Knoten hängt am 3-Knoten und ist eines der äußeren Kinder des 3-Knoten

Fall a):



Top-Down Einfügen - Fall 2a (und c)

Darstellung als Rot-Schwarz-Baum:



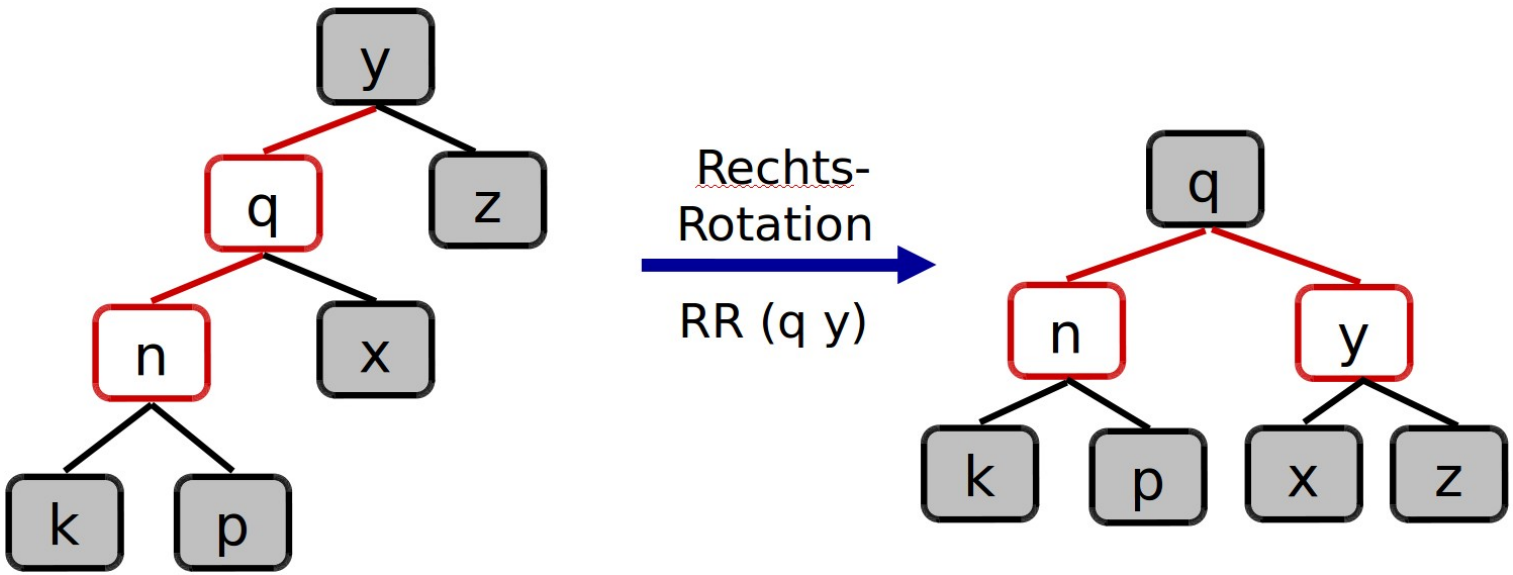
Umfärben des 4-Knoten (*k n p*) führt zu zwei aufeinanderfolgenden roten Knoten

⇒ Eigenschaft 4 der Definition verletzt

⇒ Rotation der Knoten erforderlich.

Top-Down Einfügen - Fall 2a (und c)

Rotation: Vertauschung von Knoten (Strukturtransformation), die die Sortierung erhält.

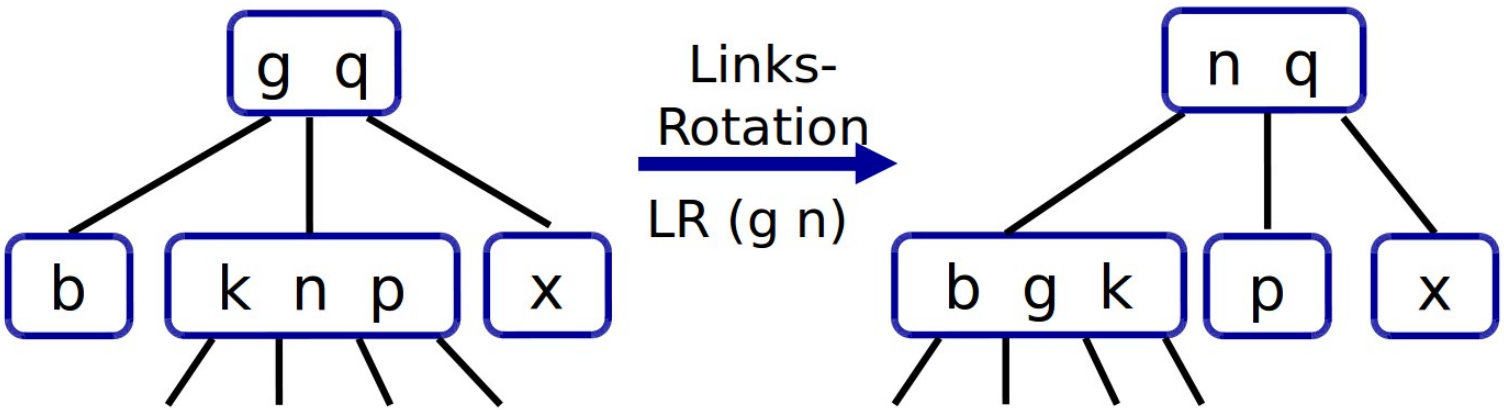


2 rote Knoten folgen aufeinander, hier beide nach links gerichtet.

Top-Down Einfügen - Fall 2b

4-Knoten ist das mittlere Kind seines Elternknoten.

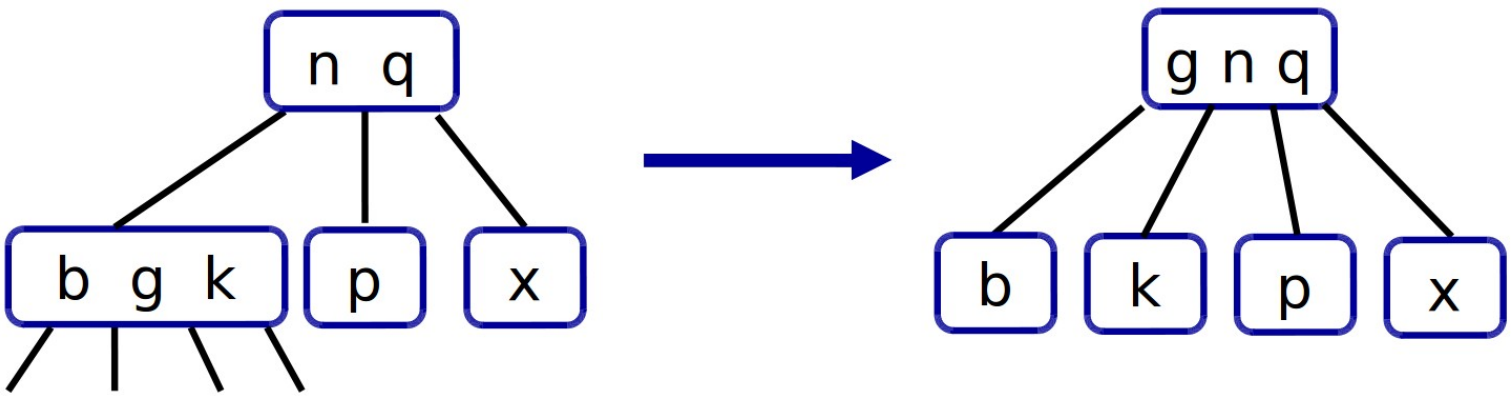
1. Rotation:



Top-Down Einfügen - Fall 2b

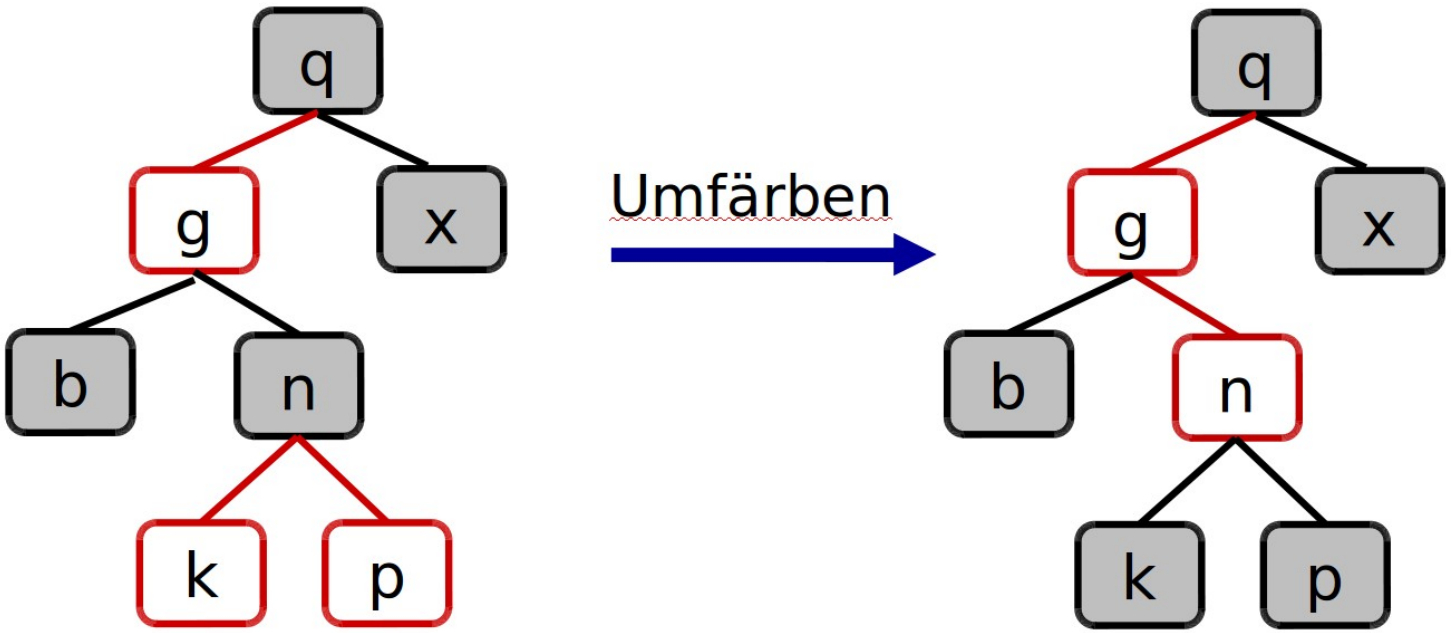
Nach der ersten Rotation mutiert Fall 2b in Fall 2a, d.h. eine erneute Rotation ist erforderlich.

2. Rotation:



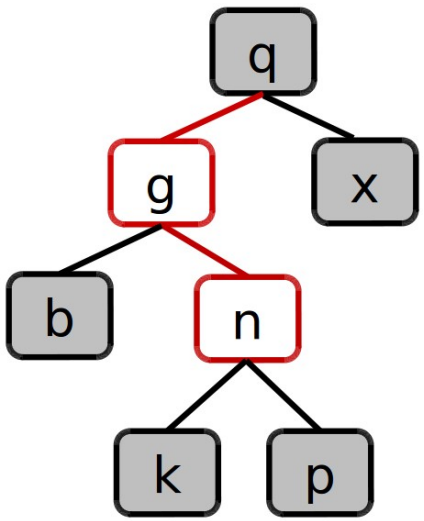
Top-Down Einfügen - Fall 2b

Darstellung als Rot-Schwarz-Baum:



Umfärben des 4-Knoten (*knp*) führt zu zwei aufeinanderfolgenden roten Knoten in der Form eines „Links-Rechts-Knicks“ ⇒ **Doppelrotation** zwischen (*q g n*)

Top-Down Einfügen - Fall 2b

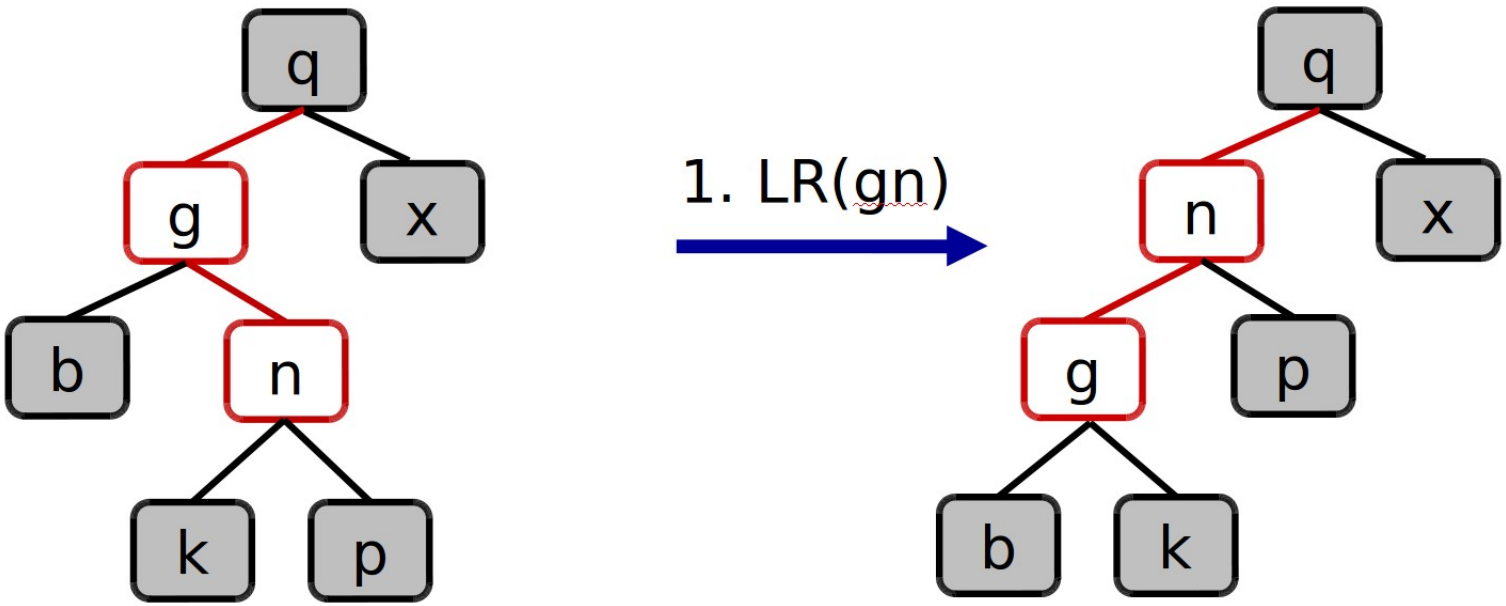


Doppelrotation zwischen (*q g n*)**:**

- Die Doppelrotation wird durch zwei Rotationen aufgelöst, sodass das mittlere Element (*n*) von den beteiligten Knoten im Baum oben steht und schwarz wird. Am Ende entsteht daraus der 4-Knoten (*g n q*) als Rot-Schwarz-Baum.
- Die erste Rotation wird auf den unteren beiden roten Knoten (*g n*) ausgeführt: Linksrotation zwischen (*g n*). Dadurch wird der Knoten *n* linker Nachbar von *q*.
- Die zweite Rotation wird dann auf den oberen beiden Knoten durchgeführt. Rechtsrotation zwischen (*q n*)

Top-Down Einfügen - Fall 2b

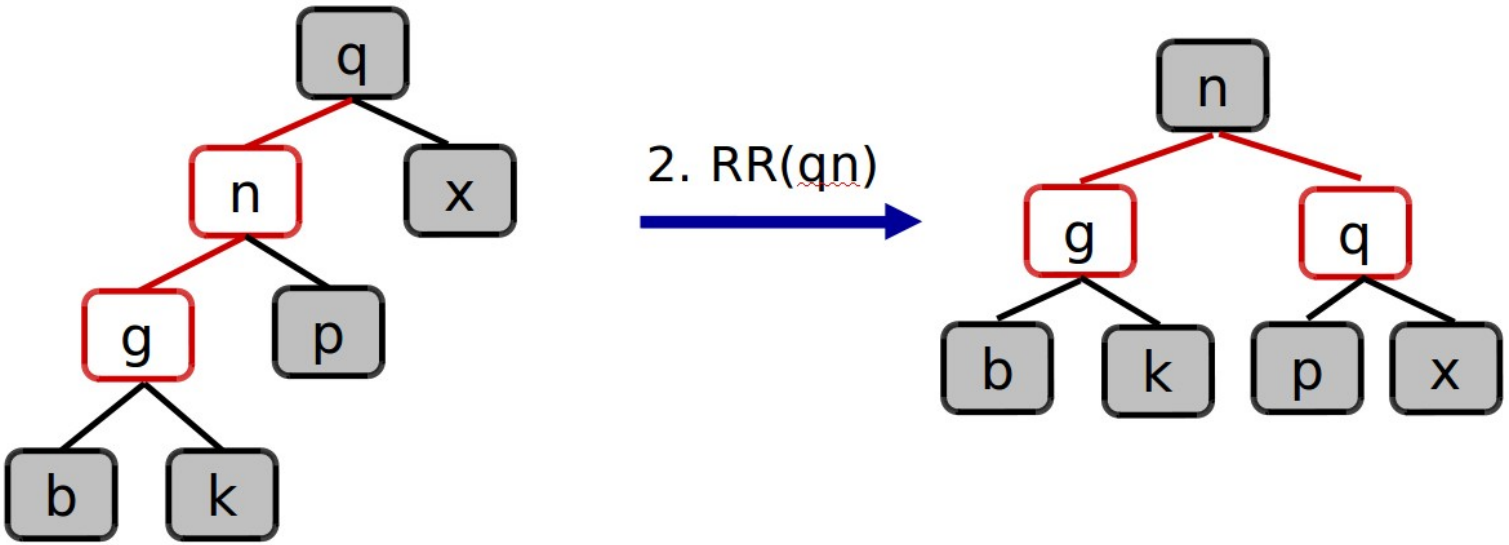
Linksrotation LR zwischen (*g n*)



Durch Linksrotation der roten Knoten mutiert der Fall 2b zu Fall 2a

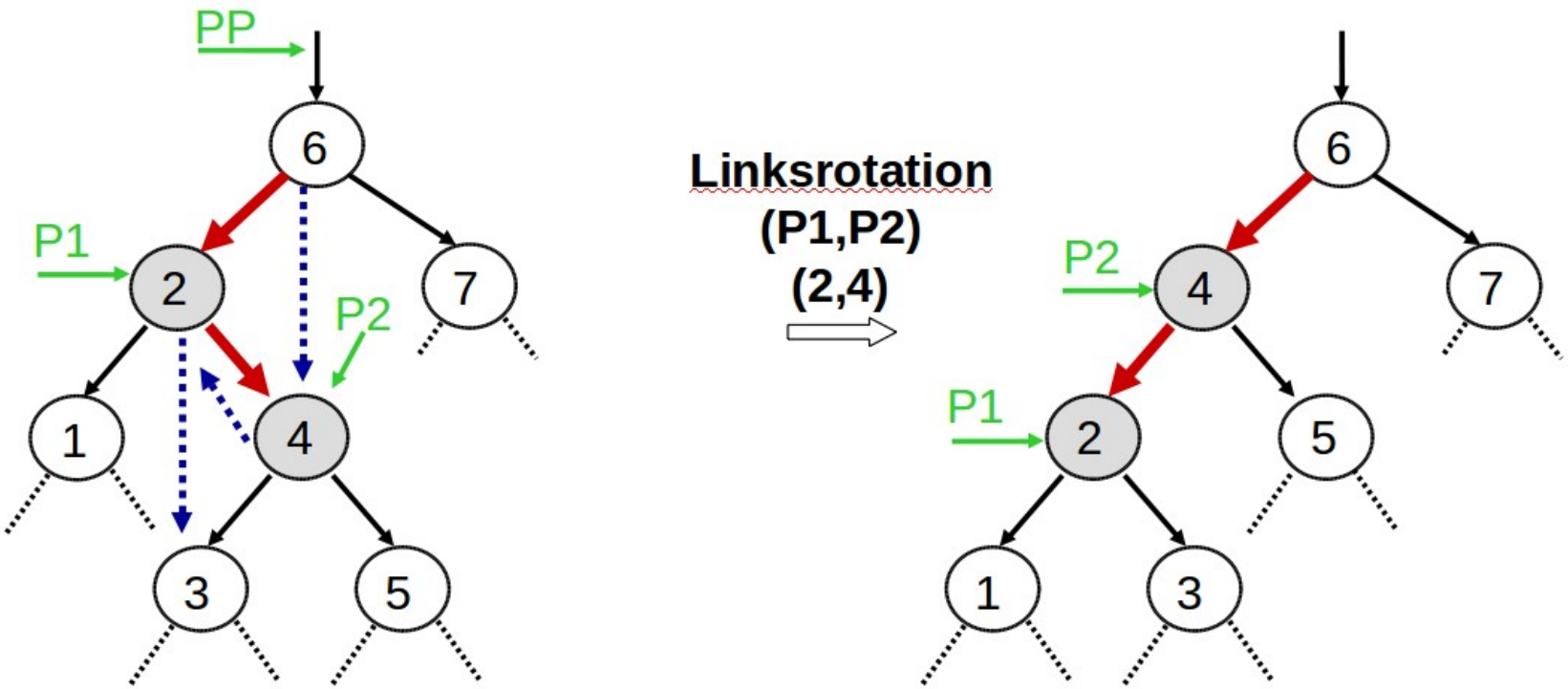
Top-Down Einfügen - Fall 2b

Rechtsrotation RR zwischen (*q n*)



Das Ergebnis der Links-Rechtsrotation der Knoten (*q g n*) ist die Erzeugung eines 4-Knoten (*g n q*) im Rot-Schwarz Baum.

Beispiel: Linksrotation



Zusammenfassung

- Ziel: BST ausgleichen beim Einfügen und Löschen
- Neue Datenstruktur: 2-3-4-Baum
- Implementierung des 2-3-4-Baumes mit der DS Rot-Schwarz-Baum
- Methoden zum Ausgleichen: Links-, Rechts-, Doppelrotation
- Wann werden Rotationen durchgeführt?

Fragen

1. Was war das Ziel der heutigen Vorlesung?
2. Welche neue Datenstrukturen haben Sie kennengelernt?
3. Wie heißt die Datenstruktur zur Implementierung des 2-3-4-Baumes?
4. Welche Methoden zum Ausgleichen kennen Sie?
5. Wann werden die Methoden zum Ausgleichen durchgeführt?
6. Nennen Sie die 5 Bedingungen für einen gültigen Rot-Schwarz-Baum.

Vielen Dank!

Prof. Ingrid Scholl
FH Aachen
Fachbereich für Elektrotechnik und Informationstechnik
Graphische Datenverarbeitung und Grundlagen der Informatik
MASKOR Institut
Eupener Straße 70
52066 Aachen
T +49 (0)241 6009-52177
F +49 (0)241 6009-52190
scholl@fh-aachen.de

www.fh-aachen.de