Algorithmen und Datenstrukturen

Master

Bäume

Inhalt

- Terminologie
- Listen (als Spezialfall)
- Binäre Bäume
- Traversieren von Bäumen

- B-Trees

Terminologie

- Baum ist ein Spezialfall eines Graphen
  - Knoten (nodes)
  - Blätter (leaves) oder Endknoten
  - Wurzel (root)
- Alle Knoten eines Baums sind durch genau einen Pfad miteinander verbunden
  - Knoten auf gleicher Ebene mit demselben Vaterknoten (parent node) werden als Geschwister (siblings) bezeichnet

Terminologie

- Geordneter Baum
  - Reihenfolge der direkten Nachfolger ist bei jedem Knoten angegeben
- Knoten lassen sich in Ebenen (levels) einteilen
  - Anzahl der Knoten auf dem Pfad bis zur Wurzel
- Höhe des Baums wird durch die höchste Ebene der Knoten festgelegt

Terminologie

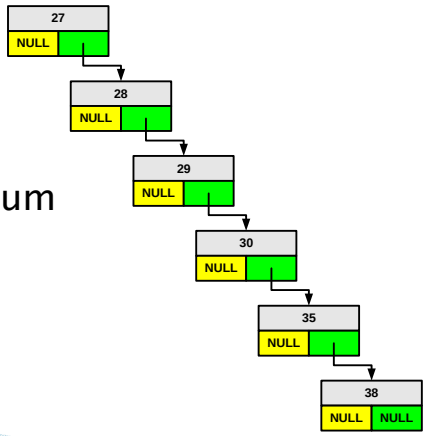
- Falls jeder Knoten im Baum eine bestimmte Anzahl n von Nachfolgern hat, spricht man von einem n-ären Baum
  - Binärer Baum (n=2)
  - Ternärer Baum (n=3)

Eigenschaften von Bäumen

- Für je zwei beliebige Knoten in einem Baum existiert genau ein Pfad, der sie verbindet.
- Ein Baum mit N Knoten hat N-1 Kanten.

Listen

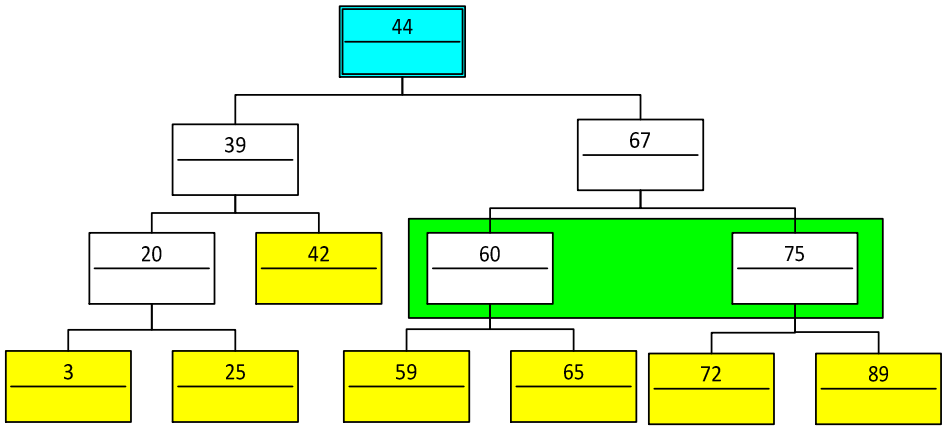
- Wenn jeder Knoten im Baum genau einen Nachfolger hat, so entsteht eine Liste
- Liste ist entarteter Baum



Eigenschaften von Binären Bäumen

- Jeder Knoten hat genau zwei Nachfolger
- Ein binärer Baum mit N inneren Knoten hat genau N+1 äußere Knoten (Blätter).
- Die Höhe eines vollen binären Baums mit N inneren Knoten beträgt etwa $\log_2 N$.
- Auffinden eines Elements erfordert O(logN) Zugriffe. Im entarteten Baum kann das bis O(N) ansteigen!

Binärer Baum



Basisoperationen

- Einfügen
  - Beginne bei Root
  - Vergleiche neuen Wert mit Knoten
    - Wenn kleiner -> gehe links weiter
    - Wenn grösser -> gehe rechts weiter
    - Wenn kein Kind mehr da, füge den neuen Wert als Knoten ein

Basisoperationen

- Löschen
  - Beginne bei Root Element
  - Suche zu entfernenden Knoten (doomed) und merke Parent-Knoten
    - Wenn doomed keine Nachfolger hat entferne Knoten und setze Zeiger im Parent auf 0
    - Wenn doomed nur einen Nachfolger hat setze Zeiger im Parent auf den Nachfolger
    - Wenn doomed zwei Nachfolger hat, bestimme kleinsten Knoten des Subbaums unter doomed und verschiebe ihn an die Stelle von doomed.

Durchlaufen des Binären Baums

- Rekursiver Algorithmus (inorder):
  - Für jeden Knoten:
    - Durchlaufe linken Ast
      - Wenn Blatt – fertig
      - Sonst Rekursion
    - Gib Knotenwert aus
    - Durchlaufe rechten Ast
      - Wenn Blatt – fertig
      - Sonst Rekursion

Durchlaufen des Binären Baums

- In Order
  - Linker Ast, Knoten, rechter Ast
- Pre Order
  - Knoten, linker Ast, rechter Ast
- Post Order
  - Linker Ast, rechter Ast, Knoten

Extremwerte

- Kleinster Knoten ist immer links unten
  - Erster Knoten In-Order
- Grösster Knoten ist immer rechts unten
  - Letzter Knoten In-Order

Probleme mit Bäumen

- Einfügen von Elementen führt zu unausgeglichenen Bäumen
- Entfernen von Elementen führt zu unausgeglichenen Bäumen

- Gleiche Baum aus, um immer optimal O(log N) zu nutzen

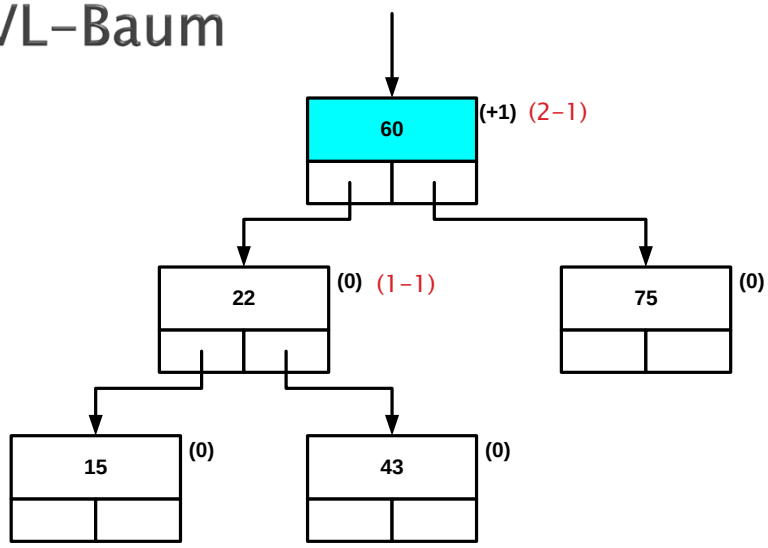
AVL-Baum

- Adelson-Velskii and Landis 1962

- Binärer Baum, der einen weiteren Faktor berücksichtigt (balance factor)
  - bf = Höhe links – Höhe rechts
  - bf > 0 -> linker Ast ist länger
  - bf < 0 -> rechter Ast ist länger

- Nach Einfügen oder Löschen müssen Elemente rotiert werden um eine möglichst gute Balance zu halten

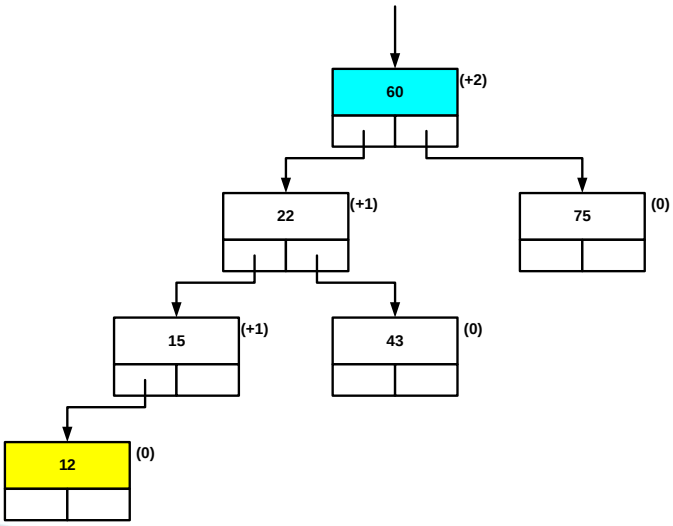
AVL-Baum



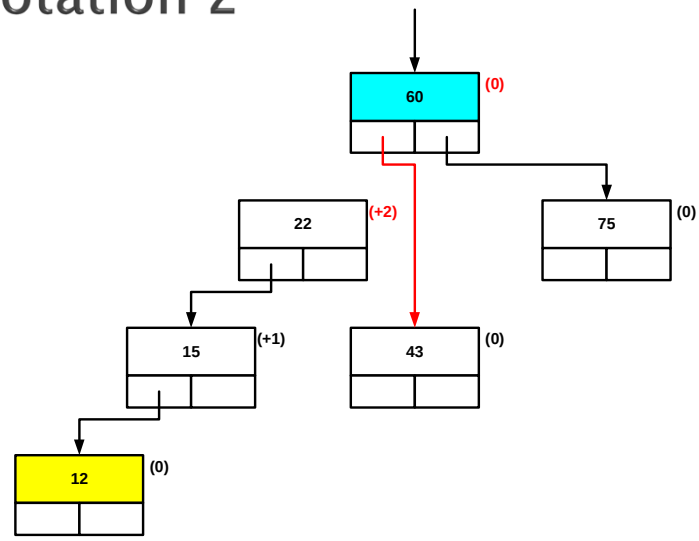
AVL-Baum

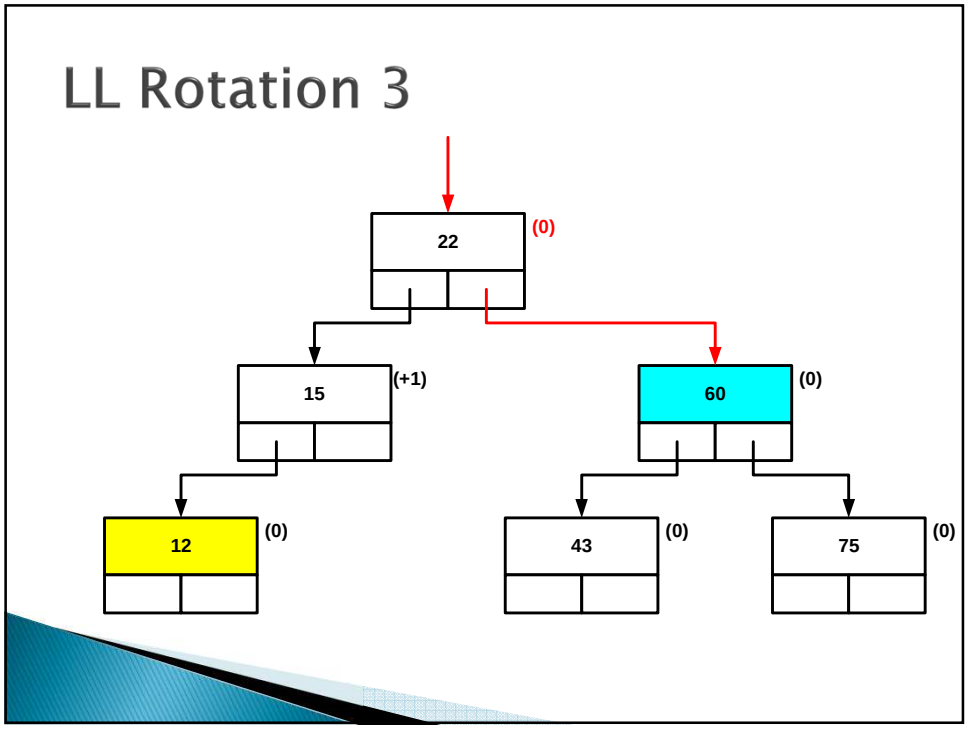
- suche nach Einfügen nähestes Element, dessen bf sich auf +/- 2 geändert hat
- 4 mögliche Rotationen
  - LL (x liegt im linken Ast des linken Astes)
  - LR (x liegt im rechten Ast des linken Astes)
  - RL (x liegt im linken Ast des rechten Astes)
  - RR (x liegt im rechten Ast des rechten Astes)

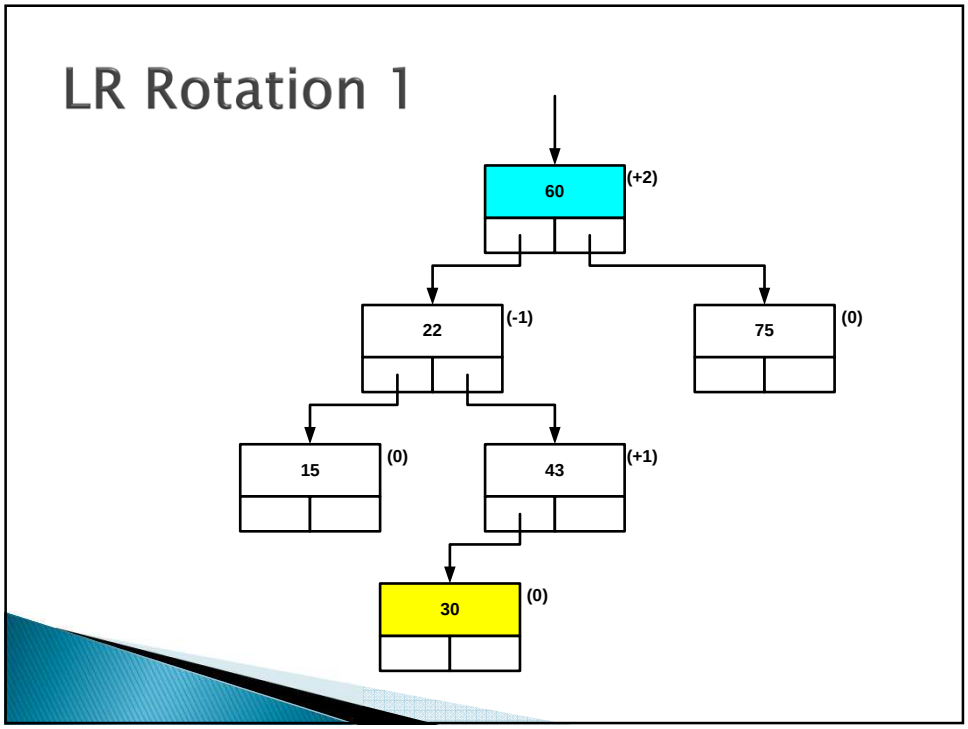
LL Rotation 1



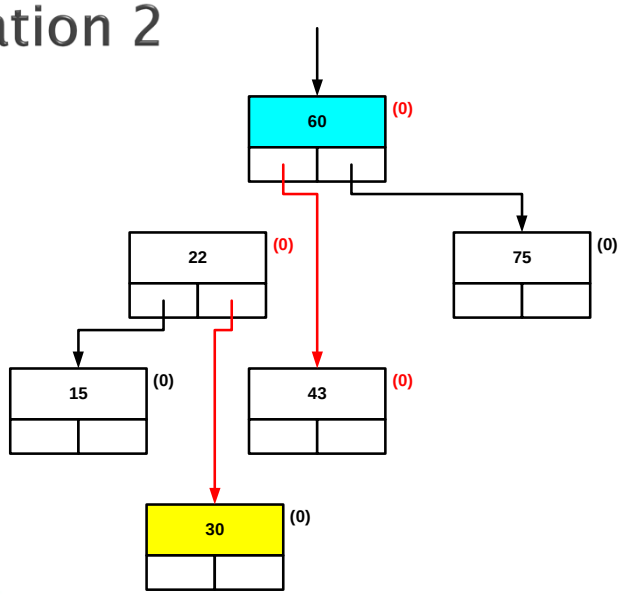
LL Rotation 2



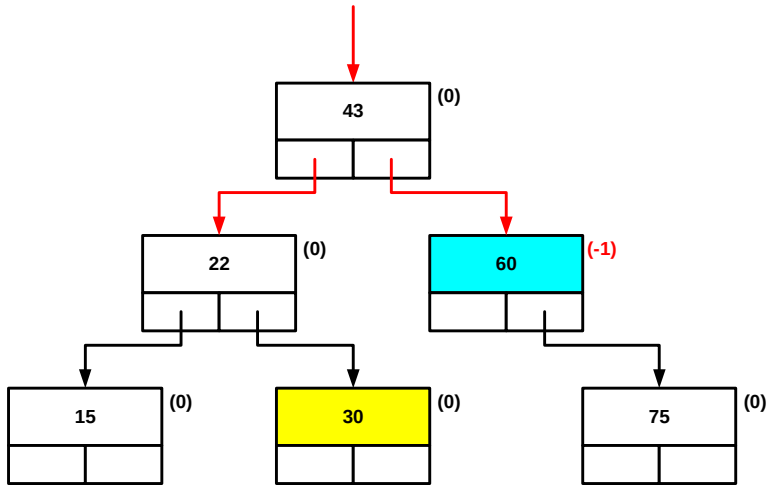

LL Rotation 3
22
(0)
15
(+1)
60
(0)
12
(0)
43
(0)
75
(0)


LR Rotation 1
60
(+2)
22
(-1)
75
(0)
15
(0)
43
(+1)
30
(0)

LR Rotation 2



LR Rotation 3



B-Baum

- allgemeine Form eines Baums

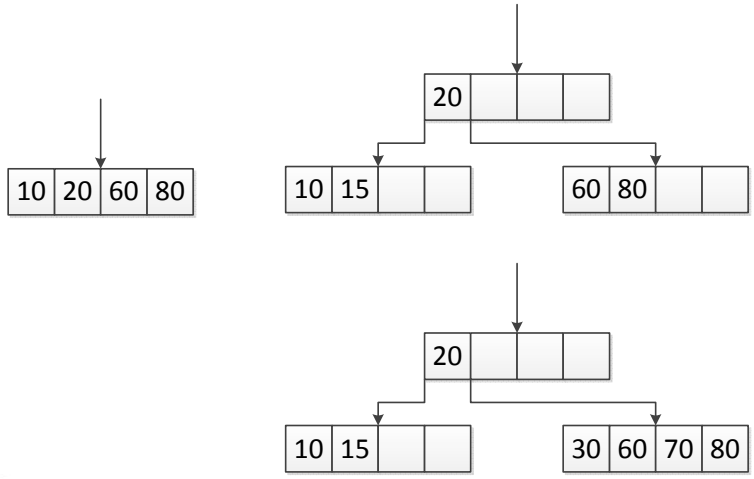
- in einer Ebene dürfen mehrere Verzweigungen auftreten
- Verwendung bei Dateien mit vielen Schlüsseleinträgen (z.B. Index bei Datenbank)
- pro Diskzugriff können gleich mehrere Schlüssel gelesen werden

B-Baum

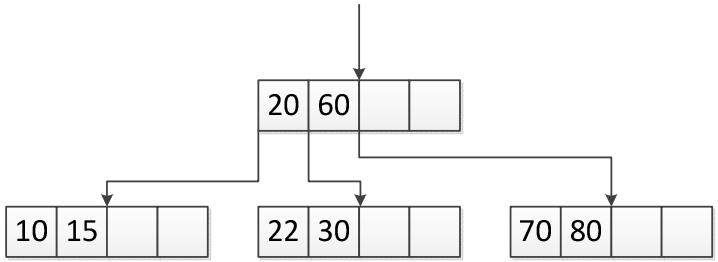
- binärer Baum enthält pro Element einen Schlüssel und zwei Äste (2-Knoten)
- 3-Knoten enthält 2 Schlüssel und 3 Äste
- ...
- N-Knoten enthält N-1 Schlüssel und N Äste

- Ausgleich wird auch hier durch Aufteilen der Knoten und Rotation hergestellt

B-Baum



B-Baum



B-Baum

- Blockweises Lesen und Schreiben begünstigt Verwendung von B-Trees in Verbindung mit Plattenspeicher
- Bessere Performance auch in Verbindung mit CPU Cache

Übung 1

- Schreibe eine AVL-Baum Klasse
  - Unittest für 4 Einfüge-Szenarien
  - Ausgabe der Baumstruktur
  - Löschen von Elementen
- Messe Zeit zum Aufbau des Baums und zur Suche von 100 Elementen mit
  - 100000 Integers in Zufallsfolge
  - 100000 Integers vorsortiert
  - Im AVL-Baum und im Binär-Baum