

Algorithmen und Datenstrukturen

B4. Bäume

Marcel Lüthi and Gabriele Röger

Universität Basel

Definitionen und Eigenschaften

Was ist ein Baum

- Struktur um Daten hierarchisch anzuordnen.

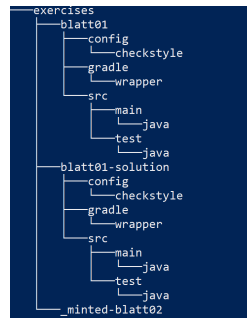
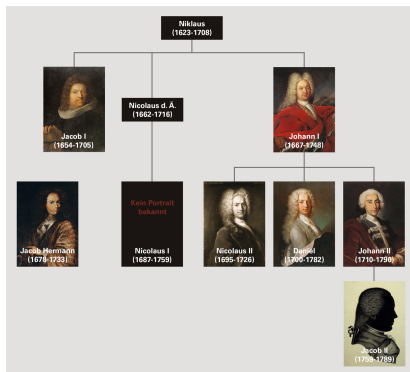


Abbildung:

<http://www.ub.unibas.ch/bernoulli/index.php/Stammbaum>

Was ist ein Baum

Rekursive Definition

Ein Baum T der Ordnung n ist

- der leere Baum,
- oder besteht aus einem Knoten (der Wurzel) sowie maximal n Bäumen (den Unterbäumen von T).

Was ist ein Baum

Rekursive Definition

Ein Baum T der Ordnung n ist

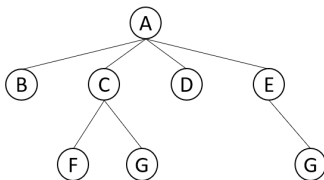
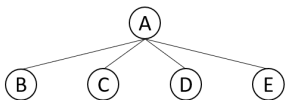
- der leere Baum,
- oder besteht aus einem Knoten (der Wurzel) sowie maximal n Bäumen (den Unterbäumen von T).

Vergleiche mit Definition von Liste:

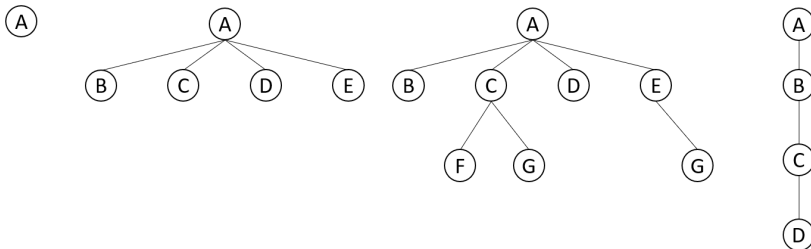
Eine Liste L ist

- die leere Liste
- oder ein Element H (Head) gefolgt von einer Liste.

Beispiele



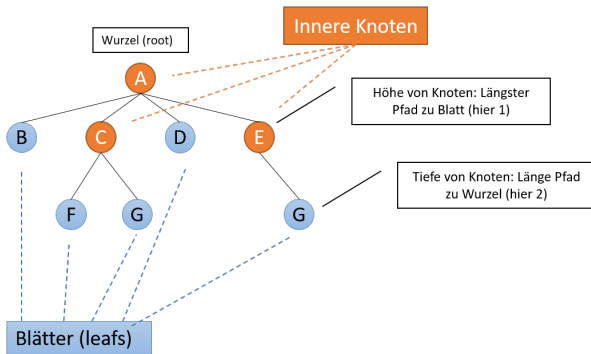
Beispiele



Eine Liste ist ein Spezialfall eines Baumes (Baum der Ordnung 1)

Terminologie

Höhe Baum
=
Höhe Wurzel
= 2



Wichtigster Spezialfall: Binärbaum

Binärbaum (Binary Tree)

Ein Binärbaum T ist

- der leere Baum
 - oder besteht aus einem Knoten (genannt Wurzel) sowie maximal 2 Binärbäumen (den Unterbäumen von T).
-
- Binärbäume haben jede Menge Anwendungen
 - Unser aktueller Fokus

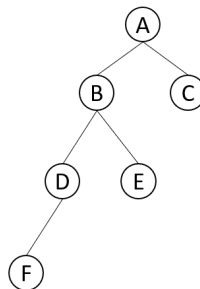


Wichtigster Spezialfall: Binärbaum

Binärbaum (Binary Tree)

Ein Binärbaum T ist

- der leere Baum
 - oder besteht aus einem Knoten (genannt Wurzel) sowie maximal 2 Binärbäumen (den Unterbäumen von T).
-
- Binärbäume haben jede Menge Anwendungen
 - Unser aktueller Fokus



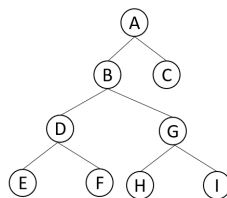
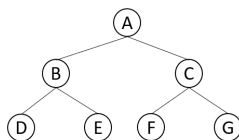
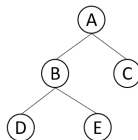
Terminologie (2)

- **Voller Binärbaum:** Jeder Knoten hat 0 oder 2 Kinder
- **Vollständiger (oder kompletter) Binärbaum:** Alle Ebenen sind vollständig gefüllt ausser evtl. die letzte Ebene wobei nur Blätter rechts fehlen dürfen.
- **Perfekter Binärbaum:** Alle internen Knoten haben genau 2 Kinder und alle Blätter sind auf der gleichen Ebene

Quiz

- Welche der folgenden Bäume sind voll, vollständig oder perfekt?
- Wie ist es mit dem leeren Baum?

Ⓐ



Höhe eines perfekten Binärbaums

Theorem

Die Höhe eines perfekten Binärbaums der Grösse N (also mit N Knoten) ist $\log_2(N + 1) - 1$.

Beweis.

- Die Anzahl Knoten N eines perfekten Baumes der Höhe h sind $N = 2^0 + 2^1 + \dots + 2^h = 2^{h+1} - 1$
- Auflösen nach h ergibt
 $\log_2(N + 1) = h + 1 \Leftrightarrow h = \log_2(N + 1) - 1$



Höhe eines vollständigen Binärbaums

Theorem

Die Höhe eines vollständigen Binärbaums der Grösse N ist $\lfloor \log_2(N) \rfloor$

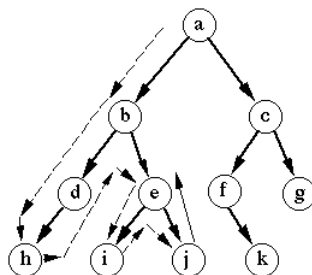
- Es stimmt für Höhe 0 (Für $N = 1$ ist $\log_2(1) = 0$)
- Die Höhe nimmt nur um 1 zu, wenn N so vergrößert wird, dass es eine Zweierpotenz wird.
 - D.h ein Knoten ist alleine auf der letzten Ebene.

Traversierung

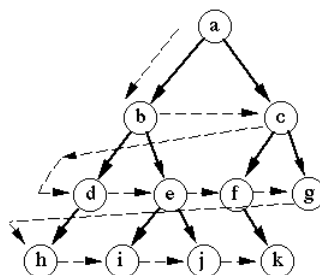
Traversierung

Breitenansatz (breadth-first-search). Eine Ebene nach dem anderen.

Tiefenansatz (depth-first-search). Zuerst in die Tiefe, dann links nach rechts.



Depth-first search



Breadth-first search

Depth-first-search Traversierung

Wir unterscheiden drei Hauptarten der DFS Traversierung:

Preorder Aktueller Knoten zuerst, danach weiter traversieren

Inorder Aktueller Knoten zwischen Traversierung von
Unterbäumen

Postorder Aktueller Knoten nach Traversierung von
Unterbäumen

Datenstruktur

Datenstruktur für Binärbaum

```
class Node[Item]:  
    item : Item  
    left: Node[Item]  
    right: Node[Item]  
  
    # Konstruktor  
    NodeTree(item : Item, left : Node[Item], right: Node[Item])
```

Datenstruktur für Binärbaum

```
class Node[Item]:  
    item : Item  
    left: Node[Item]  
    right: Node[Item]  
  
    # Konstruktor  
    NodeTree(item : Item, left : Node[Item], right: Node[Item])
```

Vergleiche mit verketteter Liste:

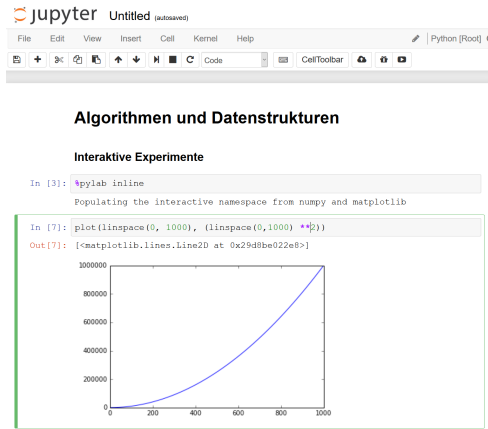
```
class Node[Item]:  
    item : Item  
    next : Node  
    Node(head : Item, next : Node[Item]) # Konstruktor
```

Rekursive Interpretation

```
class BinaryTree[Item]:  
  item  Item  
  left: BinaryTree[Item]  
  right: BinaryTree[Item]  
  
  BinaryTree(item : Item,  
             left  : BinaryTree[Item],  
             right: BinaryTree[Item]  
             )
```

- Nichts Neues: Nur neue Interpretation der Knoten (als Baum)
- Nützlich in Implementation

Implementation



IPython Notebooks: Trees.ipynb