Algorithmen und Datenstrukturen

B9. 2-3 Bäume

Marcel Lüthi and Gabriele Röger

Universität Basel

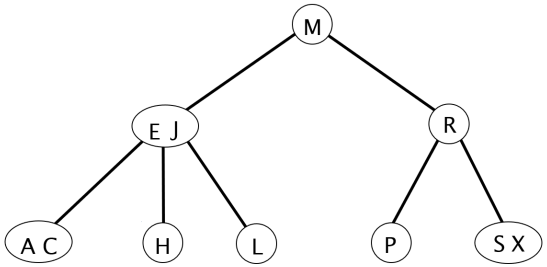
Algorithmen und Datenstrukturen
— B9. 2-3 Bäume

2-3 Bäume

Wir unterscheiden zwei Knotentypen

- 2-Knoten 1 Schlüssel, zwei Kinder
- 3-Knoten 2 Schlüssel, drei Kinder

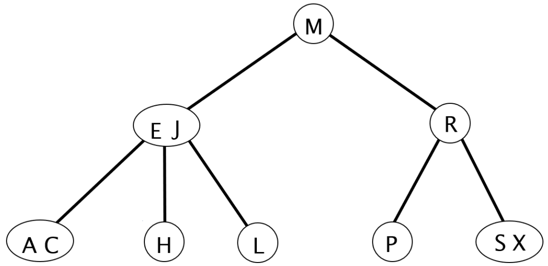
- Baum hat symmetrische Ordnung und ist perfekt balanciert.
  - Jeder Pfad von Wurzel zu Blatt hat dieselbe Länge.



Suchen in 2-3 Baum

Analog zu Binären Suchbaum

- Nutzt symmetrische Ordnung aus



Einfügen in 2-3 Baum

Einfügen in 2-Knoten auf letzter Ebene

- Neuer Schlüssel zu 2-Knoten hinzufügen. Knoten wird zu 3-Knoten.

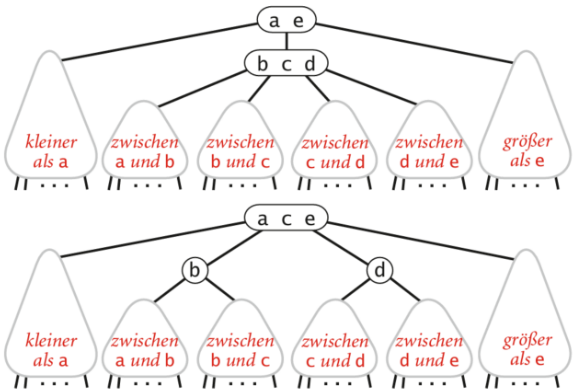
Einfügen in 2-3 Baum

Einfügen in 3-Knoten auf letzter Ebene

- Neuer Schlüssel zu 3-Knoten hinzufügen. Knoten wird temporär zu 4-Knoten.
- Mittlerer Schlüssel in Elternknoten einfügen.
- Falls nötig, rekursiv fortsetzen.

Lokale Transformationen

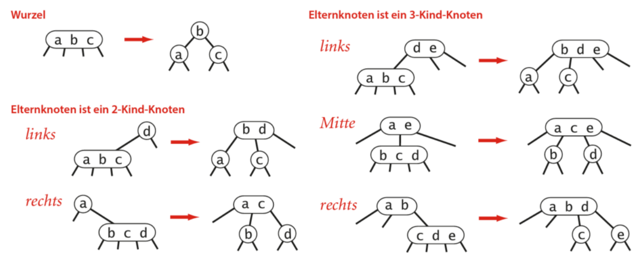
- Teilen eines 4 Knotens ist lokale Operation
  - Unterbäume nicht davon betroffen
  - Konstante Anzahl Operationen



Quelle: Abb. 3.30, Algorithmen, Wayne & Sedgewick

Globale Eigenschaften von Einfügeoperation

- Jede Operation belässt Baum perfekt balanciert.
- Ordnung der Teilbäume bleibt erhalten.



Quelle: Abb. 3.31, Algorithmen, Wayne & Sedgewick

Komplexität

Die Operationen Suchen und Einfügen in einen 2-3 Baum mit N Schlüsseln besuchen im schlechtesten Fall $\log_2(N)$ Knoten.

Übersicht

	Worst-case		Average-case	
	suchen	einfügen	suchen (hit)	einfügen
Binäre Suche	$O(\log_2(N))$	O(N)	$O(\log_2(N))$	$O(N/2)$
BST	$O(N)$	$O(N)$	$O(\log_2(N))$	$O(\log_2(N))$
2-3 Baum	$O(\log_2(N))$	$O(\log_2(N))$	$O(\log_2(N))$	$O(\log_2(N))$