Überblick

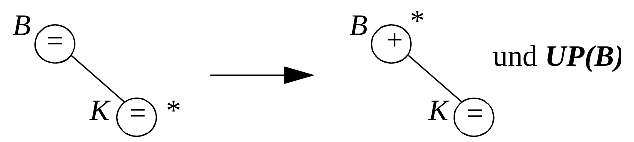
Motivation

- Die Operationen Einfügen und Löschen sind für AVL-Bäume etwas komplizierter, da Verletzungen der Struktureigenschaften auftreten können und natürlich entsprechend behandelt werden müssen.
- Um die Effizienz von $O(\log n)$ dieser Update-Operationen zu gewährleisten, müssen die Reorganisierungsmaßnahmen auf einen Pfad bis zur Wurzel beschränkt bleiben.
- Im folgenden beschreiben wir anschaulich die Einfüge- und Lösch-Operation auf AVL-Bäumen. Wir beschränken uns bei symmetrischen Fällen auf die Darstellung eines Falles.

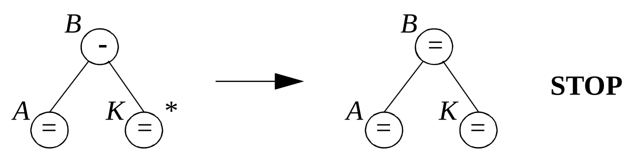
Einfügen

Der Schlüssel k wird in einen neuen Knoten K eingefügt. B bezeichnet den Vaterknoten dieses neuen Knotens. Wir unterscheiden zwei Fälle:

- Fall 1: B ist ein Blatt:



- Fall 2: B hat einen linken Sohn (symmetrisch: rechten Sohn):

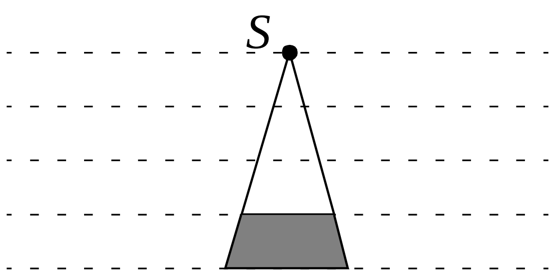


Einfügen

Die Methode $UP(S)$ wird aufgerufen für einen Knoten S mit den Eigenschaften:

- Der Baum mit Wurzel S ist in seiner Höhe um 1 gewachsen.
- Der Baum mit Wurzel S ist ein korrekter AVL-Baum.

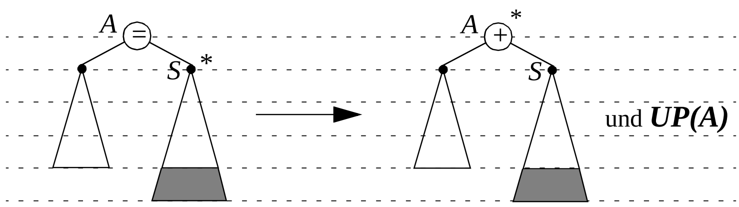
Eine mögliche Strukturverletzung ergibt sich im Vaterknoten von S durch den zu hohen Teilbaum mit Wurzel S.



Die Methode $UP(S)$ beseitigt diese Strukturverletzungen. Dabei werden folgende Fälle unterschieden:

Einfügen

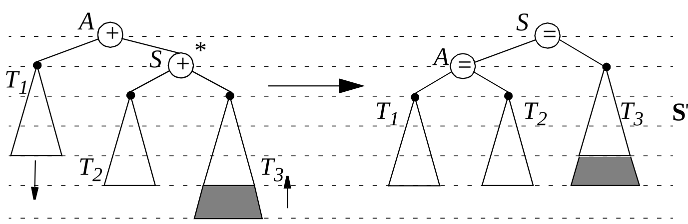
- Fall 1: der Vater von *S* hat BF “=”:
  - Fall 1.1: der Vater von *S* ist nicht die Wurzel:



  - Fall 1.2: der Vater von *S* ist die Wurzel:
    dieselbe Transformation wie Fall 1.1 und **STOP**.
- Fall 2: der Vater von *S* hat BF “+” oder “-” und *S* war die Wurzel des kürzeren Teilbaumes:
  In diesen Fällen wird der BF im Vater zu “=” und **STOP**.

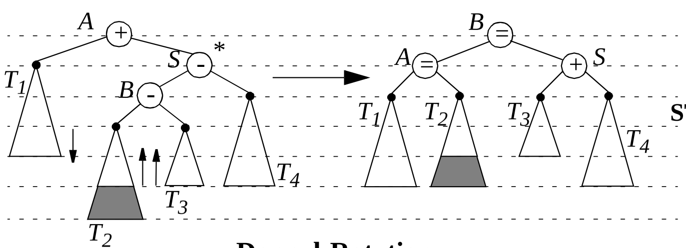
Einfügen

- Fall 3: der Vater von *S* hat BF “+” oder “-” und *S* war die Wurzel des höheren Teilbaumes:
  - Fall 3.1: der Vater *A* von *S* hat BF “+” und *S* hat BF “+”:



Rotation

  - Fall 3.2: der Vater *A* von *S* hat BF “+” und *S* hat BF “-”:
    z.B. der Sohn *B* von *S* hat BF “-”



Doppel-Rotation

der Fall *B* hat BF “+” wird analog gehandhabt.

Einfügen

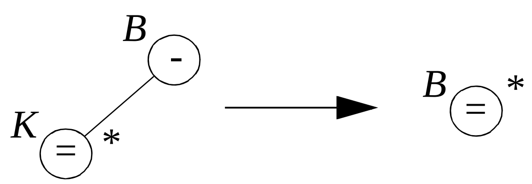
- Fall 3 (Fortsetzung)
  - Fall 3.3: der Vater *A* von *S* hat BF “-” und *S* hat BF “-”: symmetrisch zu 3.1
  - Fall 3.4: der Vater *A* von *S* hat BF “-” und *S* hat BF “+”: symmetrisch zu 3.2 (z.B. der Sohn *B* von *S* hat BF “-”)

Beim Einfügen genügt eine einzige Rotation bzw. Doppel-Rotation um eine Strukturverletzung zu beseitigen.

Löschen

Der Knoten *K* mit dem Schlüssel *k* wird entfernt. Wir unterscheiden wiederum 2 Fälle:

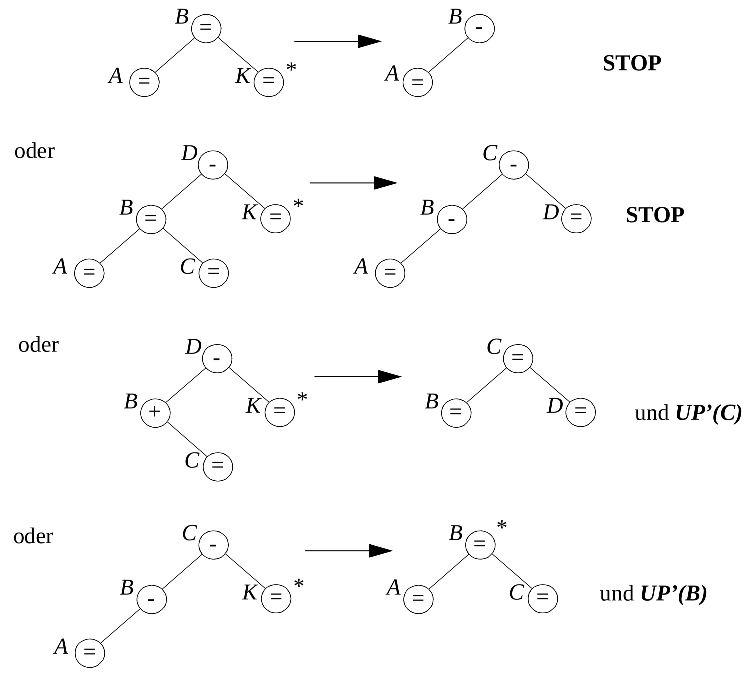
- Fall 1: *K* hat höchstens einen Sohn:
  - Fall 1.1: *K* ist ein Blatt:
    - Fall 1.1.1: *K* hat keinen Bruder:



und ***UP’(B)***

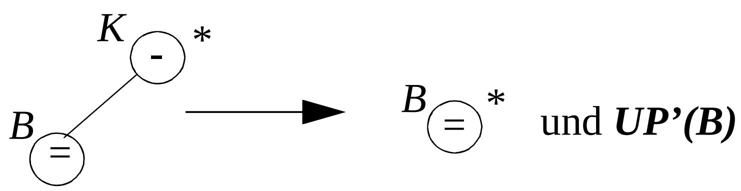
Löschen

- Fall 1 (Fortsetzung)
  - Fall 1.1 (Fortsetzung)
    - Fall 1.1.2: *K* hat einen Bruder:



Löschen

- Fall 1 (Fortsetzung)
  - Fall 1.2: *K* hat genau einen Sohn:
    - Fall 1.2.1: *K* hat einen linken Sohn:



    - Fall 1.2.2: *K* hat einen rechten Sohn: symmetrisch zu 1.2.1

Löschen

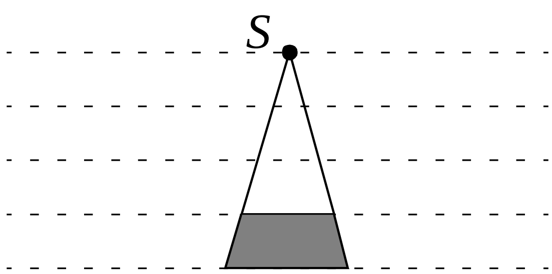
- Fall 2: K ist ein innerer Knoten (hat zwei Söhne):
  - Bestimme im AVL-Baum den *kleinsten* Schlüssel s, der *größer als* k ist. s ist in einem Halbblatt S.
  - Ersetze k durch s und entferne den Knoten S.
  - Damit ist Fall 2 auf Fall 1 zurückgeführt.

Löschen

Die Methode $UP'(S)$ wird aufgerufen für einen Knoten S mit den Eigenschaften:

- Der Baum mit Wurzel S ist in seiner Höhe um 1 reduziert worden.
- Der Baum mit Wurzel S ist ein korrekter AVL-Baum.

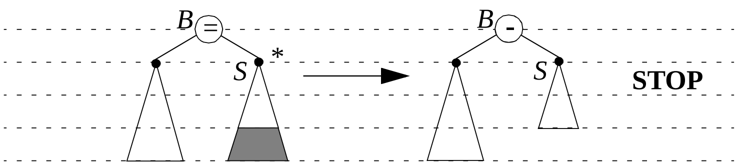
Eine mögliche Strukturverletzung ergibt sich im Vaterknoten von S durch den zu niedrigen Teilbaum mit Wurzel S.



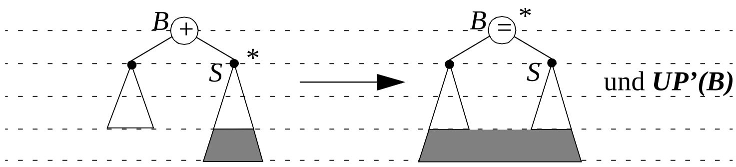
Die Methode $UP'(S)$ beseitigt diese Strukturverletzungen. Dabei werden folgende Fälle unterschieden:

Löschen

- Fall 1: der Vater von *S* hat BF “=”:

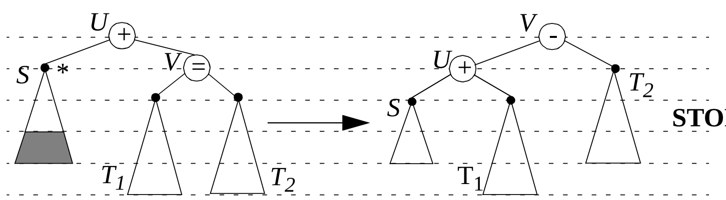


- Fall 2: der Vater von *S* hat BF “+” oder “-” und *S* ist die Wurzel des höheren Teilbaums:



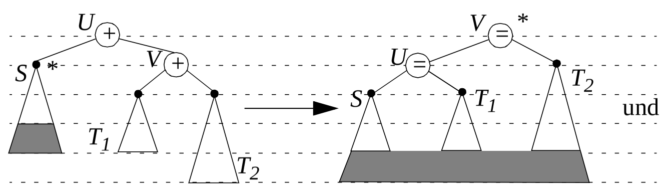
Löschen

- Fall 3: der Vater von *S* hat BF “+” oder “-” und *S* ist die Wurzel des kürzeren Teilbaums:
  - Fall 3.1: der Vater von *S* hat BF “+”:
    - Fall 3.1.1: der Bruder von *S* hat BF “=”:



Rotation

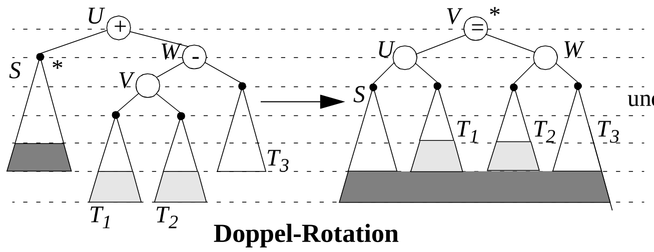
- Fall 3.1.2: der Bruder von *S* hat BF “+”:



Rotation

Löschen

- Fall 3 (Fortsetzung)
  - Fall 3.1 (Fortsetzung)
    - Fall 3.1.3: der Bruder von *S* hat BF "-":



Doppel-Rotation

Mindestens einer der beiden Bäume T_1 und T_2 hat die durch den abgeteilten Bereich angegebene Höhe.

  - Fall 3.2: der Vater von *S* hat BF "-": symmetrisch zu 3.1
- Fall 4: *S* ist die Wurzel: **STOP**.

Löschen

- Im Falle von Lösch-Operationen wird eine mögliche Strukturverletzung also nicht notwendigerweise durch eine einzige Rotation bzw. Doppelrotation beseitigt.
- Im schlechtesten Fall muss auf dem Suchpfad bottom-up vom zu enfernenden Schlüssel/Knoten bis zur Wurzel auf jedem Level eine Rotation bzw. Doppelrotation durchgeführt werden.
- Damit bilden die AVL-Bäume eine Klasse balancierter Bäume.

Überblick

Zusammenfassung

Sie kennen jetzt

- vollständig ausgeglichene binäre bzw. balancierte binäre Suchbäume und deren Vorteile bzgl. der Effizienz,
- AVL-Bäume als Klasse balancierter binärer Suchbäume.