Algorithmen und Datenstrukturen SS 2013: Gültige Konventionen

B-Baum

Entfernen: Wir betrachten für einen möglichen Ausgleich nur den rechten Bruderknoten. Falls dieser minimal gefüllt ist, wird verschmolzen, ansonsten ausgeglichen. Falls es keinen rechten Bruderknoten gibt, so betrachten wir den linken Bruderknoten.

B*-Baum

Einfügen: Wir betrachten für einen möglichen Ausgleich erst den rechten Bruderknoten. Wenn dieser bereits maximal gefüllt ist oder es keinen rechten Bruderknoten gibt, dann betrachten wir den linken Bruderknoten. Wenn dieser maximal gefüllt ist, dann verteilen wir den aktuellen Knoten zusammen mit dem rechten Bruderknoten (falls dieser existiert) auf drei Knoten, ansonsten zusammen mit dem linken Bruderknoten.

Entfernen (wie im Skript): Hat der Knoten, aus dem entfernt wird, einen linken und einen rechten Bruderknoten, so wird wenn möglich mit dem rechten, ansonsten mit dem linken Bruderknoten ausgeglichen. Falls beides nicht möglich ist, so werden die drei Knoten zu zwei Knoten zusammengefasst. Wenn der Knoten nur einen Bruderknoten (rechts oder links) hat und dieser nicht minimal gefüllt ist, so wird mit diesem Knoten ausgeglichen. Ist der Bruderknoten minimal gefüllt, so wird zunächst dieser mit dem ersten indirekten Bruderknoten ausgeglichen, dann der aktuelle mit dem Bruderknoten. Falls dies nicht möglich ist, so werden der aktuelle Knoten, der Bruderknoten und der erste indirekte Bruderknoten zu zwei Knoten zusammengefasst.

Lineares Hashing mit partiellen Erweiterungen

Wird direkt nach einer Expansion die Kontrollfunktion (Speicherplatzauslastung) immer noch überschritten, ohne dass ein weiterer Schlüssel eingefügt wurde, so wird nicht sofort wieder expandiert, sondern erst nach dem Einfügen des nächsten Schlüssels. (Siehe dazu auch die Musterlösung zu Aufgabe 9-1, Folie 4 auf der Webseite.)

Heap-Sort

Der Aufbau des initialen Heaps erfolgt von unten nach oben und für jedes Level von links nach rechts.